Chapter 17
A Survey and Taxonomy of Energy Efficient Resource Management Techniques in Platform as a Service Cloud

Sareh Fotuhi Piraghaj
The University of Melbourne, Australia

Rodrigo N. Calheiros
The University of Melbourne, Australia

Amir Vahid Dastjerdi
The University of Melbourne, Australia

Rajkumar Buyya
The University of Melbourne, Australia

ABSTRACT

The numerous advantages of cloud computing environments, including scalability, high availability, and cost effectiveness have encouraged service providers to adopt the available cloud models to offer solutions. This rise in cloud adoption, in return encourages platform providers to increase the underlying capacity of their data centers so that they can accommodate the increasing demand of new customers. Increasing the capacity and building large-scale data centers has caused a drastic growth in energy consumption of cloud environments. The energy consumption not only affects the Total Cost of Ownership but also increases the environmental footprint of data centers as CO2 emissions increases. Hence, energy and power efficiency of the data centers has become an important research area in distributed systems. In order to identify the challenges in this domain, this chapter surveys and classifies the energy efficient resource management techniques specifically focused on the PaaS cloud service models.

BACKGROUND

The numerous advantages of cloud computing environments, including cost effectiveness, on-demand scalability, and ease of management, encourage service providers to adopt them and offer solutions via cloud service models. In return, it encourages platform providers to increase the underlying capacity of their data centers to accommodate the increasing demand of new customers. One of the main drawbacks of the growth in capacity of cloud data centers is the need for more energy to power these large-scale

DOI: 10.4018/978-1-5225-0759-8.ch017

infrastructures. This drastic growth in energy consumption of cloud data centers is a major concern of cloud providers.

An average data center consumes as much energy as 25,000 households, as reported by Kaplan et al. (Kaplan, Forrest, & Kindler, 2008). This energy consumption results in increased Total Cost of Ownership (TCO) and consequently decreases the Return of Investment (ROI) of the cloud infrastructure. Apart from low ROI, energy consumption has a great impact on carbon dioxide (CO2) emissions, which are estimated to be 2% of global emissions (Buyya, Beloglazov, & Abawajy, 2010).

The energy wastage in data centers are caused by various reasons such as inefficiency in data center cooling system (S. Greenberg, Mills, Tschudi, Rumsey, & Myatt, 2006), network equipment (Heller et al., 2010), and server utilization (A. Greenberg, Hamilton, Maltz, & Patel, 2008). However, servers are still the main power consumers in a data center (A. Greenberg et al., 2008). Both the amount of work and the efficiency with which the work is performed affects the power consumption of servers (Krioukov et al., 2010). Therefore, for improving the power efficiency of data centers, the energy consumption of servers should be made more proportional to their workload.

The power proportionality is defined as the proportion of the amount of power consumed comparing to the actual workload. The power proportionality can be achieved by either decreasing the servers idle power at hardware level (Barroso & Holzle, 2007) or efficient provisioning of servers through power aware resource management policies at software level. In this chapter, we solely focus on software level and the resource management techniques utilized for decreasing energy consumption in Cloud data centers considering four different service models (depicted in Figure 1):

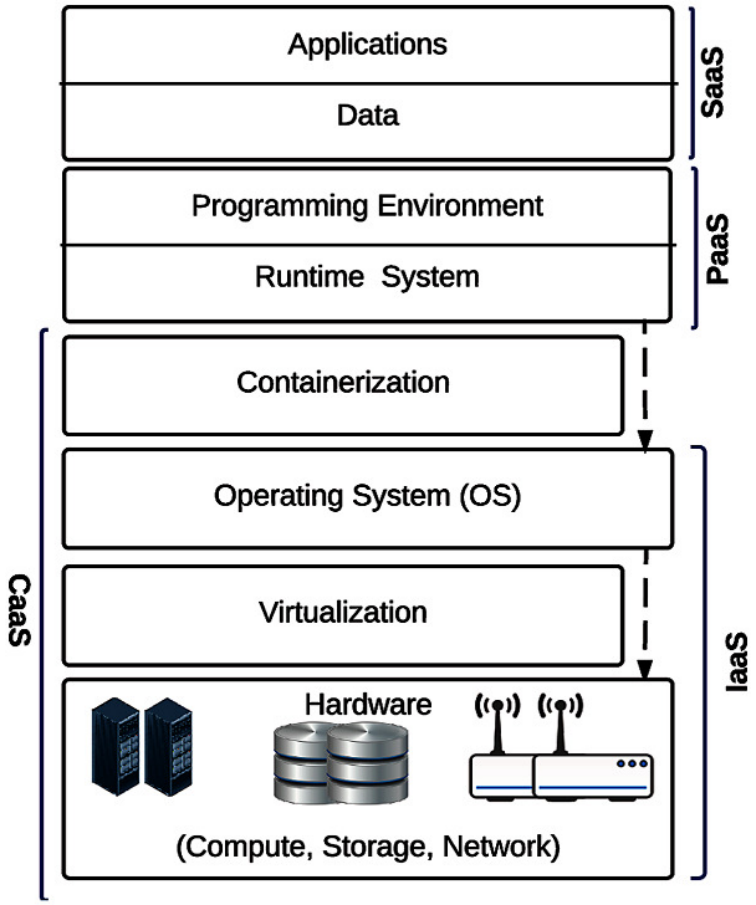
- **Infrastructure as a Service (IaaS):** In this service a consumer has the ability to provision the required resources while running and deploying arbitrary software such as operating systems and applications. Using this model consumers do not need to worry about the underlying hardware.
- **Platform as a Service (PaaS):** This model has a higher level of abstraction in comparison to the IaaS model. By offering the application-hosting environment, the consumers do not need to have any control over the underlying infrastructure including storage, processing and network.
- **Software as a Service (SaaS):** Using this service model, a consumer is able to use the provider's applications which are hosted on the Cloud. Applications are accessible through web portals. This model has also made development and testing easy for providers via having access to the software.
- **Containers as a Service (CaaS):** This service is recently introduced and lies between IaaS and PaaS. While IaaS provides virtualized compute resources and PaaS provides application specific runtime services, CaaS is the missing layer that glues these two layers together.

Among the above-mentioned service models, this chapter mostly focuses on energy efficient resource management techniques for PaaS and CaaS.

PaaS POWER-AWARE RESOURCE MANAGEMENT

There is a large body of literature investigating energy management techniques for PaaS cloud service model that provides a platform for cloud customers to develop, run, and manage their applications without worrying about the underlying infrastructure and the required software. Both kinds of virtualization namely, OS level and System level virtualization, are considered and the newly introduced CaaS model

Figure 1. The container as a service cloud service model links the PaaS and IaaS layers



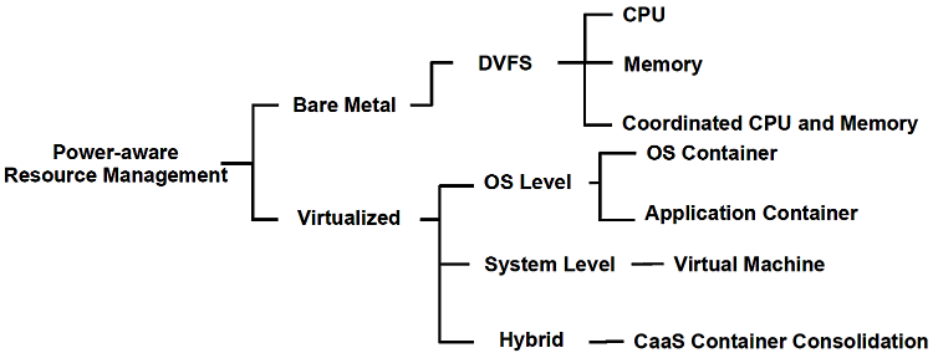
can be viewed as a form of OS level virtualization service. Since CaaS cloud model has been newly introduced, we grouped all the research with the focus on containerized (OS-level virtualized) cloud environments under the PaaS category.

The work in this area, as demonstrated Figure 2, is grouped in two major categories namely "Bare Metal, non-virtualized", and "Virtualized". The *Bare Metal* group contains the techniques in which the applications/tasks are mapped to the servers without considering virtualization technology, whereas the work investigating energy efficient techniques in a virtualized environment are all included in *Virtualized* group.

Bare Metal Environments

Servers are one of the most power-hungry elements in data centers, with CPU and memory as their main power consumers. The average power consumption of CPU and memory is reported to be 33% (Meisner, Gold, & Wenisch, 2009) and 23% (David, Fallin, Gorbatov, Hanebutte, & Mutlu, 2011) of the server's total power consumption respectively. Therefore, any improvement on processor and memory-level power

Figure 2. Power-aware PaaS resource management research breakdown



consumption would definitely reduce the total power consumption of the server, which also improves the energy efficiency of data center.

Dynamic Voltage and Frequency Scaling (DVFS) is an effective system level technique utilized both for memory and CPU in *Bare Metal* environments and it is demonstrated to improve the power consumption of these two elements considerably (Charles, Jassi, Ananth, Sadat, & Fedorova, 2009; David et al., 2011; Deng, Meisner, Ramos, Wenisch, & Bianchini, 2011; von Laszewski, Wang, Younge, & He, 2009). DVFS enables dynamic power management through varying the supply voltage or the operating frequencies of the processor and/or memory. Research in this area are summarized in Tables 2- 3.

Dynamic Voltage and Frequency Scaling of CPU

The technologies present in the market are AMD Turbo Core (AMD, 2016), Intel Turbo Boost (Charles et al., 2009), and Intel Enhanced Speed Stepping Technology (Intel, 2016), which dynamically adjust the CPU frequency and voltage according to the workload. Kim et al. (K. H. Kim, Buyya, & Kim, 2007), harnessed the DVFS capability of CPU in the proposed scheduling algorithm. DVS scheduling scheme considers the deadline of the Bag-of-Tasks applications as a constraint and the CPU frequency is adjusted so that the sub-tasks are finished by the deadline. An application made of a group of independent and identical tasks is an example of Bag-of-Task applications. DVS scheduling algorithms are provided for both time-shared and space-shared resource sharing policies. Proposed algorithm is validated through simulation and is shown to be more energy efficient when compared to the static voltage schemes.

Pietri et al. (Pietri & Sakellariou, 2014) also proposed an energy efficient scheduling algorithm utilizing the DVFS capability of CPU. The frequency of the CPU is adjusted with the objective of reducing the total energy consumption for the execution of tasks while meeting a user-specified deadline. Decreasing the overall energy consumption is considered as the objective of the algorithms, since DVFS is not always energy efficient, as scaling the CPU frequency may increase the execution time. Hence, it escalates the processors idle time. Based on the aforementioned objective, it is demonstrated that the lowest possible frequency is not always the most energy-efficient option. Therefore, the proposed approach only scales the frequency if the overall energy consumption can be minimized.

Table 2. Energy efficient research considering bare metal environment

Authors	Workload	SLA	Energy Saving	Energy Model
David et al. (David et al., 2011)	SPEC CPU2006 benchmark workload	Application slow down	DVFS on Memory component for CPU intensive work- loads	Real-system measurements along with an analytically power reduction model for memory.
Deng et al.(Deng et al., 2011)	Memory Intensive, CPU Intensive, and balanced Workloads	User defined performance degradation limit.	DVFS on memory component	A power model is proposed to include the effect of the memory
Deng et al.(Deng et al., 2012)	Memory and CPU intensive and a combination of these workloads, The workload characteristics are known beforehand	Application performance degradation limit	DVFS on both CPU and memory component	System total energy model is proposed containing both CPU and memory frequency
Kim et al. (K. H. Kim et al., 2007)	Bag-of-Tasks	Task Execution time	DVFS on CPU component	$E = E_{dynamic} + E_{static} =$ k1V2 L + k2(k1V2 L) = αV2 L
Leverich et al. (Leverich & Kozyrakis, 2010)	Hadoop's MapReduce workload	Throughput	Powering down idle nodes which are not accessed regularly, a set of nodes are considered as a backup to ensure the data availability.	Linear Power Model. (CPU only)
Lang et al. (Lang & Patel, 2010)	Workload Characteristics such as the expected resource consumption and performance goals of jobs are studied and considered as abstract meta-models	Response Time	Power down/up MR nodes to save energy in periods of low utilization.	An energy model is presented which incorporates the power drawn by both online and offline nodes during the workload execution.
Kaushik et al.(Kaushik et al., 2010)	One-month of Yahoo Hadoop's HDFS logs in a multi-tenant cluster are grouped according creation date and access rate. Two main categories are considered namely hot and cold zones.	Response Time	Energy-efficient data-placement through dividing servers into two major groups namely Hot and Cold zones. Energy can be saved through harnessing the idleness in the Cold zone.	Power models are used for the power levels, transitions times of power states and the subsystems access time including the disk, the processor and the DRAM.

Dynamic Voltage and Frequency Scaling of Memory

In addition to CPU, memory of servers also consumes a considerable amount of energy that is not proportional to the load (David et al., 2011). For memory-intensive workloads, system's memory speed is well tuned and optimized according to the peak computing power. However, there is still a place for improvement for other kinds of workloads that are less sensitive to the memory speed. For these kinds of workload, running at lower memory speed would result in less performance degradation and reduce the power consumption via running memory at a lower frequency.

David et al. (David et al., 2011) presented an approach utilizing the memory DVFS capability to tune the system's memory frequency based on the workload and consequently minimize the energy

Table 3. Energy efficient research considering bare metal environment

Authors	Workload	SLA	Energy Saving	Energy Model
Chen et al. (Y. Chen et al., 2012)	MapReduce with interactive analysis (MIA) style workloads is classified using k-means clustering algorithm.	Response time	BEEMR (Berkeley Energy Efficient MapReduce) as an energy efficient MapReduce workload manager which is inspired by studying the Facebook Hadoop workload	Linear Power Model (CPU only).
Feller et al.(Feller et al., 2015)	Hadoop benchmarks including three micro-benchmarks namely TeraGen, TeraSort, and Wikipedia data processing are studied and the approach is applicable for both Bare metal and System level.	Application's completion time.	Achieved through considering the resource boundness of the tasks along with the differences between the map and reduce tasks.	Energy metered environment is utilized (Grid5000).
Lee et al.(Lee et al., 2015)	Bare Metal and System level Workflow applications Workload. Workload Characteristics is assumed to be a prior knowledge	Makespan	Improved the resource utilization.	NA
Pietri et al.(Pietri & Sakellariou, 2014)	Scientific Workflow applications	Makespan	DVFS on CPU component	Energy consumption pf is estimated considering each processors operating frequency f through a cubic model derived in [] Pf = Pbase + Pdif * (f – fbase/fbase)3 and the total power consumption is estimated adding Pf to the power consumed when the processor is idle.
Durillo et al.(Durillo et al., 2014)	Workflow applications	Makespan	Efficient Resource Allocation	Energy consumption of task A running on multi core systems is estimated through: $c.E_{core}+E_{share}$. Here, c is the number of active cores and E_{core} is the energy consumed in active cores while executing task A. E_{shared} is the energy consumption of all the shared subsystems which are active during the task execution.

consumption. Additionally, a detailed power model is presented which quantifies dependency portions of memory power to the frequency and further proves the possibility of considerable power deduction through memory DVFS. Also, a control algorithm is proposed to tune frequency/voltage of memory considering its bandwidth utilization with the objective of minimizing performance degradation. The approach is evaluated through implementation on real hardware while SPEC CPU2006 is used to generate the workload. This work can further be extended for different types of workloads considering DVFS application for both CPU and memory components.

Deng et al. (Deng et al., 2011) introduces active lower-power modes (*MemScale*) for main memory to make it more energy proportional. In this respect, DVFS and dynamic frequency scaling (DFS) are applied on the memory controller and its channels and DRAM devices, respectively. *MemScale* is implemented as an operating system policy and, like David et al. (David et al., 2011) it identifies the

Table 1. Hardware virtualization taxonomy

Virtualization	Component	Communication with Hardware	Available Technologies
Operating System (OS)	OS Container (Lightweight VM)	System Standard Calls	LXC, OpenVZ, Linux VServer, FreeBSD Jails, Solaris zones
	Application Container		Docker, Rocket
System	Virtual Machine (VM)	Hypervisor	KVM, VMWare

DVFS/DFS mode for the memory subsystem according to the bandwidth utilization of memory. The objective of the research is also similar to the work by David et al. (David et al., 2011), which improves the energy consumption of the memory subsystem. This is important because it can reach up to 40% of the system's energy utilization (Hoelzle & Barroso, 2009). *MemScale* is evaluated through simulation considering a large set of workloads with less than 10% performance degradation while in (David et al., 2011) only one workload is studied.

Coordinated CPU and Memory DVFS

Deng et al. (Deng, Meisner, Bhattacharjee, Wenisch, & Bianchini, 2012) introduced *CoScale*, which jointly applies DVFS on memory and CPU subsystems with the objective of minimizing the systems total power consumption. *CoScale* is the first work in this area that coordinates DVFS on CPU and memory considering performance constraints. The frequency of each core and the memory bus is selected in a way that energy saving of the whole system is maximized. Therefore, the selected frequencies are not always the lowest ones.

As observed by Dhimsan et al. (Dhiman, Pusukuri, & Rosing, 2008), lowering the frequency sometimes results in more energy consumption. So *CoScale* always balances the system and component power utilization. It efficiently searches the space of available frequency settings of CPU and memory and sets the components voltage according to the selected frequencies. In this respect, the algorithm should consider $m * n * c$ possibilities in which m and c are the number of available frequency setting for memory and CPU respectively and n is the number of CPU cores. In order to accelerate the search process, a gradient-descent heuristic is proposed that iteratively estimates the frequencies of the components through the presented online models. Memory-intensive (MEM), compute-intensive (ILP), compute-memory balanced (MID), and a combination of workloads are applied as the input of the system. The results of *CoScale* is further compared with four different algorithms, namely *MemScale* (Deng et al., 2011), CPU DVFS, a fully uncoordinated, and a semi-coordinated algorithm. In the fully uncoordinated algorithm, both memory and CPU frequency are decided by their managers independently. In semi-coordinated policy, the CPU manager is aware of the degradation caused by the memory manager decision in the previous cycle through accessing overall performance slack. *CoScale* satisfies the performance target while being robust-across the search space parameter.

Table 4. Energy efficient research considering os-level virtualization

Authors	Workload	SLA	Energy Saving	Energy Model
Dong et al. (Dong et al., 2014)	Google Cluster Data (OS Container)	NA	DVFS, Container Placement	Proposed a power model named as VPC
Pandit et al. (Pandit et al., 2014)	Synthetic workload (OS Container)	NA	Container placement, Simulated Annealing is applied.	NA
Hindman et al. (Hindman et al., 2011)	Synthetic workload (OS Container)	Response Time	Resource sharing between various programming models	NA
Spicuglia et al. (Spicuglia et al., 2015)	Primary Big Data application workloads (Shark [], YARN []), and the background applications computing π. (Application Container)	Throughput	Power Capping (Considers a power constraint for running applications.)	Linear Power Model (CPU only)
Anselmi et.al (Anselmi et al., 2008)	Three Tier Application workload (Application Container)	Response Time	Efficient allocation of resources	NA
Rolia et al. (Rolia et al., 2003)	Enterprise application workload (Application Container)	NA	Efficient Allocation of Resources, Servers with fast migration ability is used	NA
Mohan et al. (Mohan Raj & Shriram, 2011)	Request arrival for web servers that follows Poisson distribution (Application Container)	Allowable pending requests for each application in the dispatcher queue	NA	Power Consumption of servers based on the number of VMS

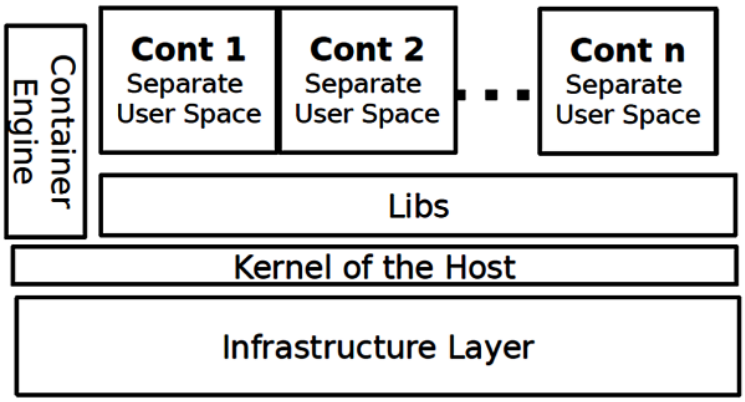
Virtualized Environments

Virtualization technology is one of the key features in cloud data centers that can improve the efficiency of hardware utilization through resource sharing, migration, and consolidation of workloads. The technology was introduced in the 1960's (Goldberg, 1974; Graziano, 2011) and exists in many levels. Of interested in this chapter, is virtualization at operating system level and at system level (Table 1).

In system-level virtualization, there exists the emulated hardware referred as "virtual machines" (VMs) that have their own operating system (OS) running on top of the host's hypervisor with independent kernels. However, on the operating system level, there exists the so called container that shares the same kernel with the host and is defined as lightweight virtual environment that provides a layer of isolation between workloads without the overhead of the hypervisor-based virtualization.

Considering these two virtualization types, techniques investigating power-aware resource management are divided into three main categories namely "Lightweight Container", "Virtual Machine", and "Hybrid". These groups are formed according to the environment in which applications execute. Therefore, the *Lightweight container* category contains techniques that assume that tasks/applications execute inside containers. In the *Virtual Machine* group, applications execute inside virtual machines.

Figure 3. Containerized virtual environment



Finally in the *Hybrid* category, applications execute inside the containers while containers are mapped on virtual machines instead of servers.

Next, we discuss these three groups with more details and explore techniques that are applied to minimize the data center energy consumption considering the characteristics of each virtualized environment. The research in this area is summarized in Table 4.

Operating System (OS) Level Virtualization (Containers)

The Platform as a Service (PaaS) model has accelerated application development and eliminated the need for administration of the underlying infrastructure. In this service model, application isolation is achieved through the utilization of containers that can run both on PMs and VMs.

Containers are the building blocks of OS-level virtualization that offer isolated virtual environments without the need for intermediate monitoring media such as hypervisors, as shown in Figure 3.The container technology of the Linux Kernel are developed separately by four different resources including OpenVZ (OpenVZ, 2016) from Parallels, Google's *cgroups* (control groups), IBM's Dpar, and namespaces (Rosen, 2013). Among those, *cgroups* and *namespaces* presented solutions for resource management and per process isolation respectively and except for the Dpar, the other three are currently used (Bottomley, 2013).

Containerization technology has been implemented on large scale by cloud companies such as Google and Facebook. Containers are beneficial for cloud providers since they can be more densely packed when compared to VMs. The other benefit of containers is that they all share the host kernel. Therefore,

Figure 4. Energy management techniques which are applied to the OS level virtualization environments

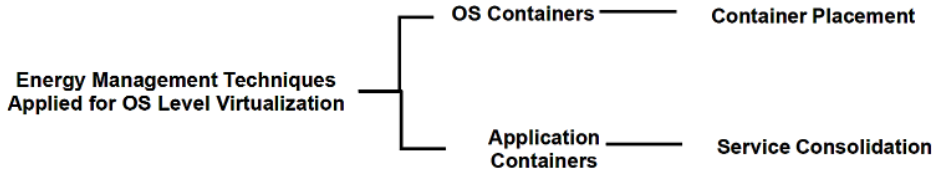
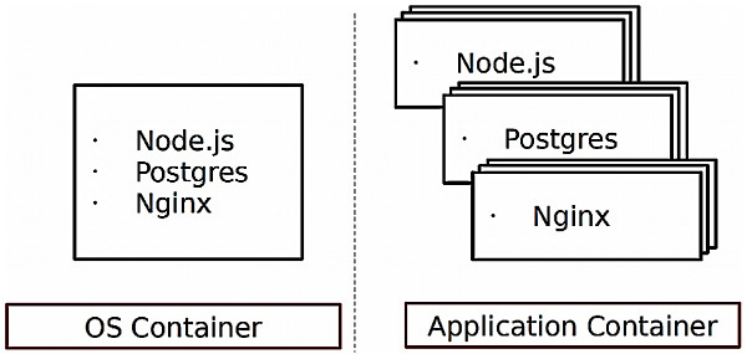


Figure 5. The differences between the Application container and the OS container for a three tier application. Application containers are implemented to run a single service and by default has layered Filesystems
Source: Nagy, 2015



the communication between containers and the hardware is performed through standard systems calls, which is much faster than hypervisor-based communication.

Operating System level virtualization or the containerization itself is categorized in two different types including OS containers and application containers and the energy management techniques which are applied to these environments are depicted in Figure 4.

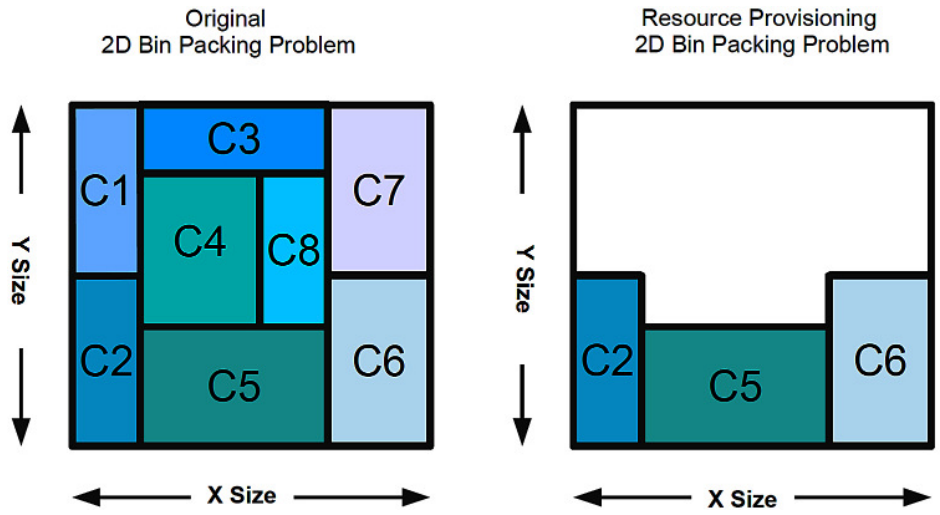
OS containers can be taught of VMs that share the kernel of the host's operating system while providing isolated user space. Various OS containers with identical or different distributions can run together on top of the host operating system as long as they are compatible with the host kernel. The shared kernel improves the utilization of resources by the containers and decreases the overhead of container's startup and shutdown. OS containers are built up on the *cgroups* and namespaces, whereas application containers are built upon the existing container technologies. Application containers are specifically designed for running one process per container. Therefore, one container is assigned for each component of the application. Application containers, as demonstrated in Figure 5, are specifically beneficial for microservice architecture in which the objective is having a distributed and multi component system that is easier to manage if anything goes wrong.

Operating System (OS) Containers

OS containers based on *cgroups* and *namespaces* provide user space isolation while sharing the kernel of the host operating system. The development in OS containers is like VMs and one can install and run applications in these containers as he runs it on a VM. Like VMs, containers are created from templates that identify the contents (Nagy, 2015). Google cluster is an example of such systems that runs all its services in containers. As stated on Google open source blog (Brewer, 2016), Google launches more than 2 billion containers per week considering all of its data centers. The container technologies that support OS containers are LXC (Container, 2016), OpenVZ, Linux VServer (VServer, 2016), FreeBSD Jails and Oracle's Solaris zones (Price & Tucker, 2004).

Energy efficient resource management techniques applied for OS container systems mostly focus on the algorithms for initial placement of the OS containers. In this respect, Dong et al. (Dong, Zhuang, &

Figure 6. The difference between the original bin packing problem and its variation for the resource allocation
Source: Pandit et al., 2014



Rojas-Cessa, 2014) proposed a greedy OS container placement scheme, the most efficient server first or MESF, that allocates containers to the most energy efficient machines first. For each container, the most energy efficient machine is the server that shows the least rise in its energy consumption while hosting the container. Simulation results using an actual set of Google cluster data as task input and machine set show that the proposed MESF scheme can significantly improve the energy consumption as compared to the Least Allocated Server First (LASF) and random scheduling schemes. In addition, a new perspective on evaluating the energy consumption of a cloud data center is provided considering resource requirement of tasks along with task deadlines and servers' energy profiles.

Pandit et al. (Pandit, Chattopadhyay, Chattopadhyay, & Chaki, 2014) also explored the problem of efficient resource allocation focusing on the initial placement of containers. The problem is modeled utilizing a variation of multi-dimensional bin packing. CPU, memory, network and storage of PMs are all considered as each dimension of the problem. In a general n-dimensional bin-packing problem, there exist n sub-bins of different sizes that must be filled with objects. The resource allocation problem is different from the general form, since if any sub-bin of a bin reaches its capacity (e.g. CPU), then the bin is considered full while in the original problem this is not the case. Figure 6 demonstrates this difference. In order to design an efficient resource allocation algorithm, Pandit et al. (Pandit et al., 2014) applied Simulated annealing (SA). SA is a technique used to find optimal or sub-optimal solution for NP Hard problems such as the bin packing problem and it is often applied for discrete search space. The proposed resource allocation algorithm is demonstrated to be more efficient in terms of resource utilization when compared to the commonly used First Come First Serve (FCFS) allocation policy.

OS containers are also utilized in Mesos (Hindman et al., 2011) to provide the required isolation for workload. Mesos platform enables sharing commodity clusters between cluster computing frameworks with different programming models. The main objective of Mesos is efficient utilization of resources through sharing and also avoiding data replication for each framework. Hindman et al. (Hindman et al., 2011) proposed a two-level scheduling for the Mesos platform called *resource offers*. For the first level

of the scheduling, Mesos identifies the amount of required resources for each framework. The second level scheduling is performed by the scheduler of each framework, therefore the scheduler has the ability to accept or reject the resources while deciding about the placement of the tasks. Mesos used Linux Containers and Solaris technologies for the workload isolation. The framework is tested through applying both CPU and IO-intensive workloads derived from the statistics of Facebook cloud backend traces. The studied workloads are derived from the applications that are developed utilizing both Hadoop and MPI programming model. Results show that Mesos is highly scalable and fault tolerant and can improve the resource utilization with less than 4% overhead.

Application Containers

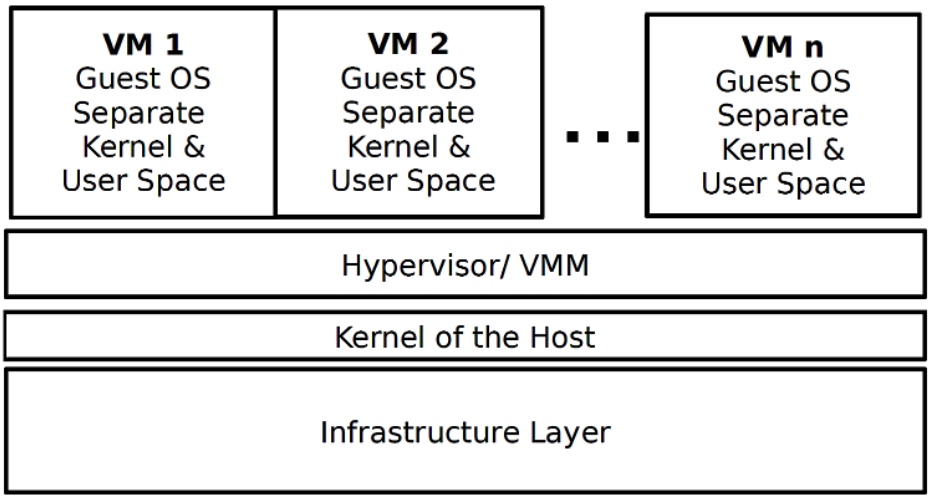
Contrary to OS containers that run multiple processes and services, application containers are dedicated to a single process and are built upon OS containers. The single process in each container is the process that runs the residing application (Nagy, 2015). Application containers can be considered a new revolution in the cloud era since containers are lightweight, easier to configure and manage, and can decrease the start-up time considerably. Docker(Merkel, 2014) and Rocket (Rocket, 2016) are examples of application containers. These containers are the building block of modern PaaS. Regular provisioning and de-provisioning of these containers, this happens during the auto-scaling, along with their unpredictable workloads results in cloud resource wastage and consequently more energy consumption. Therefore, like OS containers, designing optimal placement algorithms is the major challenge for container-based cloud providers.

Containers are fast to deploy because of their low overhead. Therefore, to simplify the development of applications, Spicuglia et al. (Spicuglia, Chen, Birke, & Binder, 2015) proposed OptiCA in which applications execute inside containers. The aim of the proposed approach is to achieve the desired performance for any given power and capacity constraints of each processor core. Although the focus in OptiCA is mainly on effective resource sharing across containers under resource constraints, it still reduces power consumption through considering energy as one of the constraints.

Anselmi et.al (Anselmi, Amaldi, & Cremonesi, 2008) investigated the Service Consolidation Problem (SCP) for multi-tier applications with the objective of minimizing the number of required servers while satisfying the Quality of Service defined by applications' response time. For modeling the data center, queueing networks theory is utilized since it is capable of capturing the performance behavior of service systems. A number of linear and non-linear optimization server consolidation problems are defined and solved through a number of heuristics. Heuristics are chosen as they solve the optimization problems in a shorter amount of time with a considerable accuracy when compared to the standard Integer Linear Programming (ILP) techniques. This work solves SCP and finds the best data-center configuration with the least cost while satisfying the required end-to-end response time of applications.

In the same direction, Rolia et al. (Rolia, Andrzejak, & Arlitt, 2003) investigated the SCP problem with the objective of minimizing the number of required servers through application consolidation. Enterprise applications are studied and their resource utilization is characterized. Like Anselmi et al. (Anselmi et al., 2008), linear integer programming is considered as one of the solutions and ILP is further compared with the genetic algorithms. The techniques are validated through a case study considering the workload of 41 servers. Results show that the linear integer programming model outperforms the genetic algorithm in terms of the required computation with a satisfactory accuracy in estimating the

Figure 7. System level virtualization energy efficient management techniques



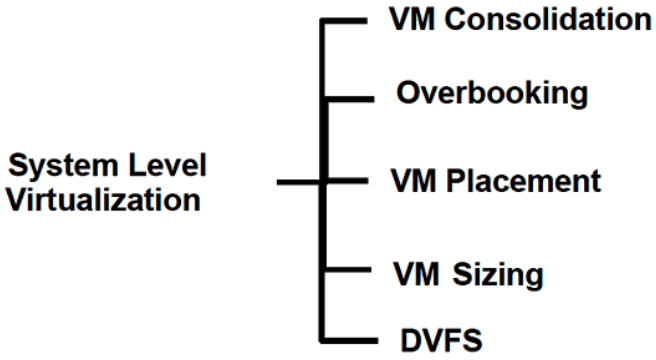
resource utilization. However, the proposed technique is not evaluated for large-scale data centers containing thousands of servers.

Mohan Raj et al. (Mohan Raj & Shriram, 2011) also focused on minimizing the energy consumption of the data center through consolidation of applications on physical machines (PM). An end-to-end Service-Level Agreement (SLA)-aware energy efficient strategy is presented in (Mohan Raj & Shriram, 2011) and the main objective is having an strategic workload scheduling for maximizing energy efficiency of the data center. Tasks are consolidated on virtual machines so that the number of active servers is reduced. Contrary to the previous discussed works (Anselmi et al., 2008; Rolia et al., 2003) synthetic workloads following the Poisson distribution are applied for the simulations to model the web server workloads. Containers are placed on the PMs that utilize the least energy rise. SLA is maintained through a control theoretic method and the requests of applications are accepted considering the SLA along with the data center capacity. The presented model is a queue-based routing approach and Holt-Winters forecasting formula is utilized for improving the SLA through decreasing the cost incurred by the times system waits for a PM to startup or to shut down. The proposed algorithm is also applicable in virtualized environments, where applications execute on virtual machines instead of directly on PMs.

System-Level Virtualization (Virtual Machines)

The virtual machine's idea originated from simulated environments offered for software development when testing on real hardware was unfeasible (Goldberg, 1974). In this respect, a specific environment was simulated considering the required processor, memory, and I/O devices. Later, this idea was improved to develop efficient simulators to provide copies of a server on itself. The improvement is done so that the program running on each copy can be executed directly on the hardware without requiring any software interpretation. These copies (simulated environments) are referred to as virtual machine systems, and the simulated software is referred as the virtual machine monitor (manager) (VMM) or the hypervisor (Goldberg, 1974). This kind of virtualization is referred as system-level virtualization.

Figure 8. System-level virtualization energy efficient management techniques



In system-level virtualization, the VM communications happens through the hypervisor with more overhead than the OS standard calls for containers. However, communicating through VMM offers a stronger layer of isolation and is more secure than containers (Nagy, 2015). In addition, system-level virtualization enables VMs with any type of OS to be installed on a host (Figure 7). Moreover, this technology enables consolidating virtual machines (VM) on physical machines (PMs) to reach higher and efficient utilization levels of PMs. The virtualization technology also improves the deployment time, and the operating costs.

As depicted in Figure 8, energy management technique applied for system-level virtualization are categorized into five groups namely virtual machine consolidation, overbooking, VM placement, VM sizing, and DVFS. The research in this area is summarized in Table 5 - 7. In the rest of this section, we discuss these techniques with more details.

VM CONSOLIDATION

The hypervisor technology enables consolidation of virtual machines on physical servers. There is a vast body of literature investigating VM consolidation algorithms that can improve the energy consumption of data centers. The consolidation problem can be divided into three main sub-problems which are depicted in Figure 9. The techniques are grouped according to the sub problem it investigates.

Techniques Investigating Migration Triggers (When to Migrate?)

Virtual machine consolidation is shown to be an effective way to minimize the energy consumption of cloud data centers. However, identifying the right time to trigger migration is crucial especially when the host is overloaded. This ensures a certain level of Quality of Service (QoS).

Gmach et al. (Gmach, Rolia, Cherkasova, & Kemper, 2009) proposed an energy-efficient reactive migration controller that identifies situations in which the hosts are determined overloaded or underloaded. The overload and underload detection is defined when the server's CPU and memory utilization goes beyond or under a given fix threshold respectively. The same approach is applied in (A. Beloglazov, 2010) and the effect of these two thresholds on the overall data center energy consumption and SLA violations is studied. 30% and 70% is shown to be the efficient underload and overload threshold considering the

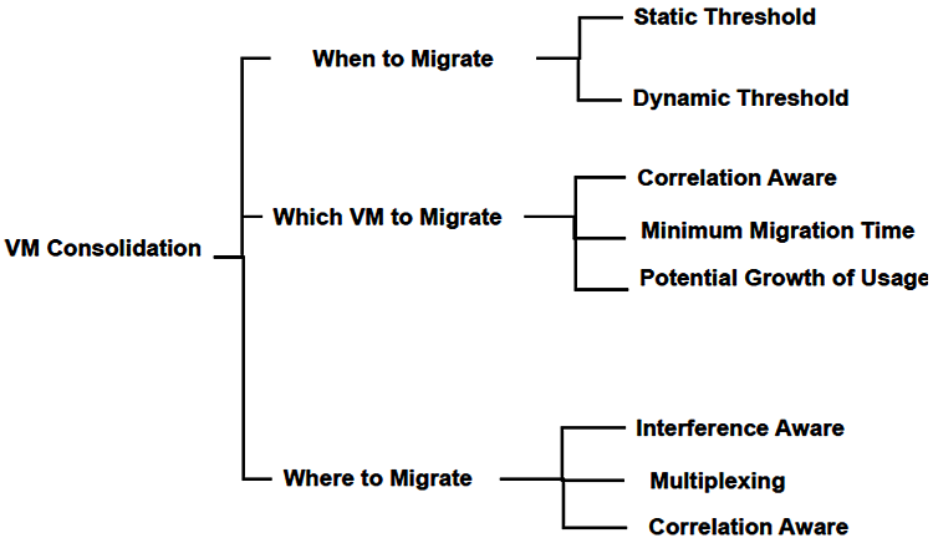
Table 5. Energy efficient research considering system-level virtualization

Authors	Workload	SLA	Energy Saving	Energy Model
Urgaonkar et al. (Urgaonkar et al., 2010)	Synthetic Workloads that follow a uniform random distribution is studied. The proposed approach is independent of workload prediction and its statistics as it uses the queueing information to learn and adapt to unpredictable changes in the workload pattern	Throughput	DVFS (on CPU component)	The power-frequency relationship is presented as a quadratic model $P(f) = P_{min} + \alpha(f - f_{min})^2$.
Gmach et al.(Gmach et al., 2009)	Enterprise application workloads analysis is utilized for estimating the under-load and over-load situations for hosts	Response Time	VM Consolidation (When to migrate)	Linear Power Model (CPU only)
Beloglazov et al. (A. Beloglazov & R. Buyya, 2010)	PlanetLab Worklaod	The time during which host experience is identified overloaded	VM Consolidation (What triggers migration with fixed Under-load and Over-load thresholds)	Linear Power Model (CPU only)
Beloglazov et al. (Anton Beloglazov & Rajkumar Buyya, 2010)	PlanetLab Workload	The time during which host experience is identified overloaded	VM Consolidation (What triggers migration with automatic under-load and overload detection algorithms along with two VM selection approaches)	Linear Power Model (CPU only)
Beloglazov et al. (Anton Beloglazov & Buyya, 2012)	PlanetLab Workload, The approach is independent of workload including stationary and non-stationary workloads	A QoS goal as an input of consolidation algorithm	VM Consolidation (What triggers migration with optimally adjusting the overload threshold and selecting the right VM to migrate)	Linear Power Model (CPU only)
Meng et.al (Meng et al., 2010)	VM workloads of a commercial data center	Response time (Considering a performance constraint for each VM)	VM Consolidation (Where to migrate considering the Statistical Multiplexing approach)	NA

total energy consumption and average SLA violations. Contrary to Gmach et al. (Gmach et al., 2009), the proposed approach (A. Beloglazov, 2010) is not dependent on the type of workload.

Beloglazov et al. (Anton Beloglazov & Rajkumar Buyya, 2010) improved the aforementioned approach (A. Beloglazov, 2010) so that the under-load and over-load thresholds are automatically adjusted. The previous approach for triggering the migration is modified since fixed values for thresholds are not suitable for cloud environments in which the workload's behavior is unknown and dynamic. The automation is performed through statistical analysis of the historical data from virtual machines workload. CPU utilization of the host is assumed to follow the t-distribution so that the sample mean and the standard deviation of the distribution can be used for determining the overload thresholds of each host. However, only one underload threshold is defined for the whole system. The adoptive approach shows a considerable improvement in terms of the QoS when compared to the fixed thresholds while it still saves energy.

Figure 9. The consolidation sub problems which need to be answered for a general consolidation problem



Cloud providers should be able to ensure the Quality of Service (QoS) that they have promised to costumers. In the consolidation process, this QoS might be degraded because of the hosts being overloaded. In this respect, Beloglazov et al. (Anton Beloglazov & Buyya, 2012) proposed a technique for host overload detection that ensures QoS while saving energy. The proposed approach can find the optimal solution for the overload detection problem considering any known stationary workload. The main objective is maximizing the intermigration time considering a given QoS goal based on a Markov chain model. In order to handle the nonstationary workloads which are unknown, a heuristic-based approach is presented that utilizes the Multisize Sliding Window technique for workload estimation. The algorithm is validated through simulations considering PlanetLab VMs traces as the input workload. The technique is proven to provide up to 88% of the performance of the optimal offline algorithm.

Techniques for Choosing VMs to Migrate (What to Migrate?)

When migration is triggered, the second step is selecting the appropriate virtual machine to migrate. For under-load hosts, it is clear that all the VMs should be migrated so that the host can be shut down or put in a lower power state. For overloaded hosts only a couple of VMs are needed to be migrated so that the host is no longer overloaded.

Beloglazov et al. (Anton Beloglazov & Buyya, 2012), investigated the problem of VM selection policies in the consolidation process. Three different VM selection algorithms are studied namely random selection (RS), maximum correlation (MC), and the Minimum Migration Time (MMT) policies. The RS policy chooses VMs randomly until the host is not overloaded anymore. The MC policy chooses the VM with the maximum correlated workload with the other co-located VMs. The MMT policy selects the VM with the least migration time.

The performance of these policies are validated through simulation and the VM types are derived from Amazon EC2 instance Types (AWS, 2016). PlanetLab's workload (PlanetLab, 2016) is used as

Table 6. Energy efficient research considering system-level virtualization

Authors	Workload	SLA	Energy Saving	Energy Model
Moreno et al. (Moreno, Yang, et al., 2013)	Google Workload is Classified based on task resource Usage patterns	QoS is insured through placing VMs based on their interference with the co-located VMs	VM Consolidation Where to migrate VMs? Interference Aware VM Placement	Linear Power Model (CPU only)
Caglar et al. (Caglar et al., 2013)	Google Workload	Response Time	VM Consolidation Where to migrate VMs? Interference Aware VM Placement	NA
Chen (M. Chen et al., 2011) et al.	The workload data of 5,415 servers from ten different companies is characterized via fitting distributions	A probabilistic function that contains the probability of a host being overloaded	VM Consolidation Where to migrate VMs? Statistical Multiplexing VM Placement	NA
Verma et al. (Verma et al., 2009)	Enterprise application workload from the production data center of a multi-national Fortune Global 500 company. A detail analysis of the workload is presented including the correlation between the workloads.	Number of time instances that an application demand is more than the server's capacity	VM Consolidation Where to migrate VMs? Statistical Multiplexing VM Placement	The power models are derived from actual measurements on the servers (Power vs CPU utilization).
Calheiros et al.(Calheiros & Buyya, 2014)	Bag of tasks applications considering CPU Intensive tasks	User defined deadlines	DVFS (CPU only)	The power consumption of each host is calculated considering the contribution of each CPU core.
Laszewski et al. (von Laszewski et al., 2009)	Worldwide LHC Computing Grid (WLCG) workload	Throughput	DVFS (CPU only)	Energy is estimated considering CPU frequency
Kim et.al (Jungsoo Kim et al., 2013)	Analyzed the characteristics of scale-out applications obtained from a data center.	Virtual machines are provisioned according to their peak load.	DVFS Efficient resource allocation placing VMs based on correlation analysis	Energy Model in [].
Tomas et al. (Tomas et al., 2014)	Real-life interactive workloads and non-interactive batch applications	Response Time	Overbooking resources	NA

the CPU utilization of the VMs. Since the applied workload is for single core VMs, the VM types are all assumed to be single-core and the other resources are normalized accordingly.

The performance of the algorithms is compared considering the total energy consumption by the physical servers of the data center and the SLA violations. Considering the results, the MMT selection policy outperforms the MC and RS policies in terms of the total joint power consumption and the SLA violations. Minimization of the VM migration time is proven to be more important than the correlation between the VMs allocated to a host.

Beloglazov et al. (Anton Beloglazov & Rajkumar Buyya, 2010) also compared two other VM selection policies including Minimization of Migrations (MM) and Highest Potential Growth (HPG) with the RS. The MM algorithm selects the least number of VMs to migrate with the objective of decreasing

Table 7. Energy efficient research considering system-level virtualization

Authors	Workload	SLA	Energy Saving	Energy Model
Klein et al. (Klein et al., 2014)	RUBiS and RUBBoS	Response time and throughput	Overbooking resources	NA
Forestiero et al. (Forestiero et al., 2014)	Logs of Eco4Cloud company www.eco4cloud.com	Avoiding any sites from being overloaded	Consolidating workloads on a multi-site platform (Geo-distributed cloud environment) Where to migrate?	NA
Khosravi et al.(Khosravi et al., 2013)	Lublin-Feitelson workload model is employed to generate the Web application and Bag-of-Tasks workload.	NA	Efficient placement of VMs	Modeled considering the frequency of CPU.
Assuncao et al. (Assuncao et al., 2012)	747 VM request workloads are classified.	Makes sure that most of the workloads can fit into selected templates	Efficient allocation of resources Decreasing the number of required templates	NA

the migration overhead. HPG chooses VMs with the lowest CPU usage compared to their requested amount, aiming at reducing the SLA violations through avoiding the total potential increase. It is shown that MM, which reduces the number of migrations, outperforms the other two algorithms in terms of the SLA violations and data center energy consumption.

TECHNIQUES INVESTIGATING MIGRATION DESTINATION (WHERE TO MIGRATE?)

When the migration is triggered and the virtual machines are selected for migration, it is the time to find a new destination for the selected VMs. Here, we describe techniques considered for finding new placement/destination for migrating virtual machines.

Interference-Aware VM Placement Algorithms

Virtualization improves resource utilization efficiency and consequently the energy consumption of cloud data centers by enabling multi-tenant environments in which diverse workload types can exist together. VM consolidation and resource overbooking improve energy savings in cloud environments. However, overbooking might affect the performance of virtual machines that are co-located on each server VM (Nathuji, Kansal, & Ghaffarkhah, 2010). The high-competitions for resource incurred between co-hosted VMs and the resource sharing nature of virtualized environment might cause performance degradation and more energy consumption. The degradation effect of co-located VMs on the performance of each other's applications on the same VM is known as *performance interference* phenomenon.

Moreno et al. (Moreno, Yang, Xu, & Wo, 2013) investigated the impact of this phenomena on the energy efficiency in cloud data centers. The problem is formulated for the virtual machine placement and is modeled utilizing the Google cloud backend traces to leverage cloud workload heterogeneity. Google tasks are grouped according to their CPU, memory and length. Three types referred as small,

medium and large are extracted through applying K-means clustering algorithm on the 18th day of the trace that has the highest submission rate. The VM placement decision is made considering the current performance interference level of each server. The interference aware VM placement algorithm is compared, via simulation, with the Google FCFS algorithm and shown to reduce the interference by almost 27.5% while saving around 15% of energy consumption.

As mentioned previously, the performance interference might degrade the QoS for real time cloud applications and consequently result in SLA violations and the placement/packing of VMs play an important role on the performance interference, since it is dependent on the workload of the co-located VMs. In this respect, Caglar et al. (Caglar, Shekhar, & Gokhale, 2013) presented an online VM placement technique included in the hALT (harmony of Living Together) middleware. hALT takes into account workload characteristics of the VMs along with the performance interference for finding placement for each VM. Machine Learning techniques are used for the online placement and the system is trained utilizing the results from an offline workload characterization. The presented framework contains three main parts, namely Virtual machine classifier, neural networks, and the decision making placement. A brief analysis of the Google cloud backend traces is presented. The analysis of the 3 days of the traces is utilized for training the classifier.

Each task in the Google traces is considered as a VM. CPU utilization, memory, and the CPI of tasks are used for the classification purpose. CPI attribute shows the `Cycle Per Instruction' metric and is used as a performance metric since it can well present the response time for compute-intensive applications (Zhang et al., 2013). Therefore, tasks that utilize more than 25% of the CPU and are compute intensive are considered for evaluation of the framework. Decrease CPI results in better performance, a result that had been demonstrated by the previous study.

Back propagation-based artificial neural networks (ANN) (Hecht-Nielsen, 1992) is used as the classifier that predicts the performance interference level. This is used to determine the best placement of the VM. The ANN is trained using the VM utilization patterns of each class of VMs, which is defined by the k-means clustering algorithm. The number of VM classes is estimated based on the maximum Silhouette value (Kaufman & Rousseeuw, 2009; Rousseeuw, 1987) and is determined to be 6 for the studied data set. The effect of the performance interference on the energy consumption is not investigated. The other drawback of the work is when the VM migration is triggered the ANN should run for every server in the data center, which may cause delays and overhead for large data centers. This can be avoided through new techniques in the search process.

Multiplexing Placement Algorithms

The other technique that is widely applied to find the new placement for selected VMs is Statistical Multiplexing. Multiplexing means sharing a resource between users with the objective of increasing the bandwidth utilization. This method has been applied to a variety of concepts including MPEG transport stream for digital TV (Haskell, Puri, & Netravali, 1997), UDP and TCP (Forouzan, 2002) protocols.

Meng et.al (Meng et al., 2010) applied the Statistical Multiplexing concept to VM placement in the server resources are multiplexed to host more VMs. In the proposed approach, called joint-VM provisioning, multiple VMs are consolidated together according to their workload patterns. Statistical multiplexing enables the VM to borrow resources from its co-allocated VMs while it is experiencing its workload spikes. In order to satisfy QoS requirements, a performance constraint is defined for each VM. This constraint ensures the required capacity for a VM to satisfy a specific level of performance

for its hosted application. Three different policies are proposed for defining the performance constraint, selecting the co-located VMs, and estimating the aggregated resource demand of multiplexed VMs with complementary workloads. The VM workloads of a commercial data center are applied for evaluation purposes and the results show that the joint-VM provisioning is considerably efficient in terms of the energy consumption.

Chen et al. (M. Chen et al., 2011) investigated the problem of VM placement with the focus on consolidating more VMs on servers. The VM placement is formulated as a stochastic bin packing problem and VMs are packed according to their effective size (ES), which is defined considering Statistical Multiplexing principles. This principle takes into account the factors that might affect the servers aggregated resource demand on which the VM is placed. The effective size is originated from the idea of effective bandwidth; however it is extended to consider the correlation of VM workloads. In this respect, the ES of a VM is affected by its own demand and its co-located VMs considering the correlation coefficient. The proposed VM placement algorithm with the order of O (1) is applied to find the best destination for VMs. Poisson and normal distributions are considered for the VM workloads. The system is also validated through simulation applying a real cloud workload trace. The effective sizing technique adds around 10% to 23% more energy saving than a generic consolidation algorithm. The optimization is performed considering only one dimension, which is CPU demand of VMs.

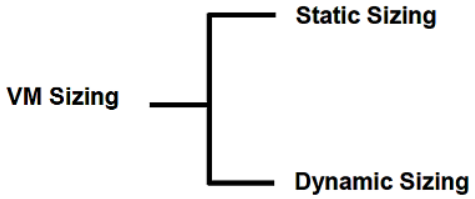
Correlation Aware VM Placement

Verma et al. (Verma, Dasgupta, Nayak, De, & Kothari, 2009) are among the first researchers to take into account correlation between workloads of co-allocated VMs in the proposed consolidation approach. The idea is initiated from a detail study of an enterprise server workload, the distribution of the utilization and spikes of the workload while considering workloads statistic metrics including the percentiles and average. According to the analysis, average is not a suitable candidate for sizing the applications since the tail of the distribution of the utilization does not decay quickly for most of the studied servers. Therefore, if the sizing is performed based on the average, it might result in QoS degradation. However, the 90-percentile and the cross correlation is shown to be fairly stable and consequently are the best metrics for application sizing purpose. Two correlation-aware placement algorithms, Correlation Based Placement (CBP) and Peak Clustering based Placement (PCP), are proposed considering the insights from the workload characterization. The placement algorithms are implemented as a part of a consolidation planning tool and further evaluated utilizing traces from a live production data center. PCP achieves more energy savings than PCB and also improves the QoS through an extra metric to ensure that co-allocated workloads' peak do not lead to violations.

Similarly, Meng et al. (Meng et al., 2010) also utilized correlation of VMs in their proposed joint-VM provisioning approach. In this approach, multiple VMs are consolidated in a way that the underutilized resources of one VM can be used by the co-located VM at its peak.

Quality of Service (QoS) is important in a cloud environment especially for scale-out applications (Ferdman et al., 2012) such as MapReduce (Dean & Ghemawat, 2008) and web searches. Therefore, Kim et.al (Jungsoo Kim, Ruggiero, Atienza, & Lederberger, 2013) investigated the VM consolidation problem concentrating on the aforementioned applications. Scale-out applications are different from HPC workloads in terms of the workload variance resulted from their user-interactive nature. This high variance is caused by external factors such as number of users and makes these workloads less predict-

Figure 10. VM sizing techniques categorized in two major groups including static and dynamic sizing



able. The other difference is that scale-out applications are latency sensitive and should maintain users expectations and consequently satisfy the SLA.

In order to save power consumption, DVFS is considered in conjunction with the VM consolidation. The voltage and the frequency are defined considering the correlation between the VMs' workloads which ensures that the expected QoS is achieved. The approach is verified through real implementation of distributed web search applications. The approach is also validated for large scale cloud workloads and compared with a correlation aware placement approach (Verma et al., 2009). The comparison shows that this approach outperforms the aforementioned technique by 13.7% and 15.6% in terms of the energy saving and QoS improvements.

Overbooking

Overbooking is an admission control method to improve resource utilization in cloud data centers. In general, cloud users overestimate the VM size they need to avoid risk of resource shortage. This provides the opportunity for providers to include an overbooking strategy (Tomas, Klein, Tordsson, & Hernandez-Rodriguez, 2014) in their admission control system to accept a new user based on anticipated resource utilization and not on the requested amount. Overbooking strategies mostly rely on load prediction techniques and manage the tradeoff between maximizing resource utilization and minimizing performance degradation and SLA violation.

Based on the resources considered for overbooking, research works can be classified into two main categories. The first category (He, Ye, Fu, & Elnikety, 2012; Jinhan Kim, Elnikety, He, Hwang, & Ren, 2013) only considers CPU and the second category (Tomas et al., 2014; Tomás & Tordsson, 2014) considers I/O and memory along with CPU. Commonly, after the overbooking phase, the majority of approaches (Hu et al., 2013; Svärd, Hudzia, Tordsson, & Elmroth, 2011) mitigate the risk of overbooking by dealing with overload of VMs on a limited number of servers. However, when the data center is overloaded, such techniques are no longer effective.

One way to deal with such challenges (which is of interest for PaaS provider) is to collect the statistics regarding application performance metrics and then, based on the priority of application and users, degrade user experience and reduce utilization of resources. There are a number of application-aware approaches proposed in the literature (He et al., 2012; Jinhan Kim et al., 2013). However, they are application-specific and only consider CPU. To this end, Klein et al. proposed brownout (Klein, Maggio, & Hernández-Rodriguez, 2014), a programming paradigm that suits cloud environments and considers CPU, IO and memory. In a PaaS environment, brownout is integrated to an application in three phases. In the first phase, the application owner (with the incentive of receiving discount on service cost) reveals which part of the hosted application can be considered non-compulsory. This part of application can

be discarded to decrease the resource requirements. In the second phase, brownout decides how often the non-compulsory computation can be discarded. Finally, in the last stage, when there is not enough capacity, brownout lessens the number of requests served with the non-compulsory part.

VM Placement

Among resource management policies, the initial placement of VMs plays an important role in the overall data center performance and energy consumption. A strategic placement of VMs can further improve the system overhead through decreasing the required number of migrations. Kabir et al. (Kabir, Shoja, & Ganti, 2014) investigated the issue assuming a hierarchical structure as a favoured deployment model for cloud service provider that consists of cloud, cluster, and hosts. This model helps in appropriately managing the geographical distributed infrastructure to achieve scalability. This hierarchical structure needs a VM placement approach that smartly provides cloud cluster, and node selection mechanisms to minimize resource fragmentation and improve energy efficiency.

In general, the placement strategies can be categorized into two classes, namely centralized and hierarchical. Khosravi et al. (Khosravi, Garg, & Buyya, 2013) proposed a centralized VM placement algorithm for distributed cloud data centers with the objective of minimizing both power consumption and carbon footprint. An information system that has the updated status regarding cloud, cluster, and host utilization is considered that enables centralized decision making and resource optimization. They considered distributed data centers with diverse carbon footprint rates and PUE values and provided a comprehensive comparison on energy efficiency of different combinations of bin-packing heuristics. They concluded that the proposed approach called energy and carbon-efficient (ECE) VM placement saves up to 45% carbon footprint and 20% of power consumption in data centers.

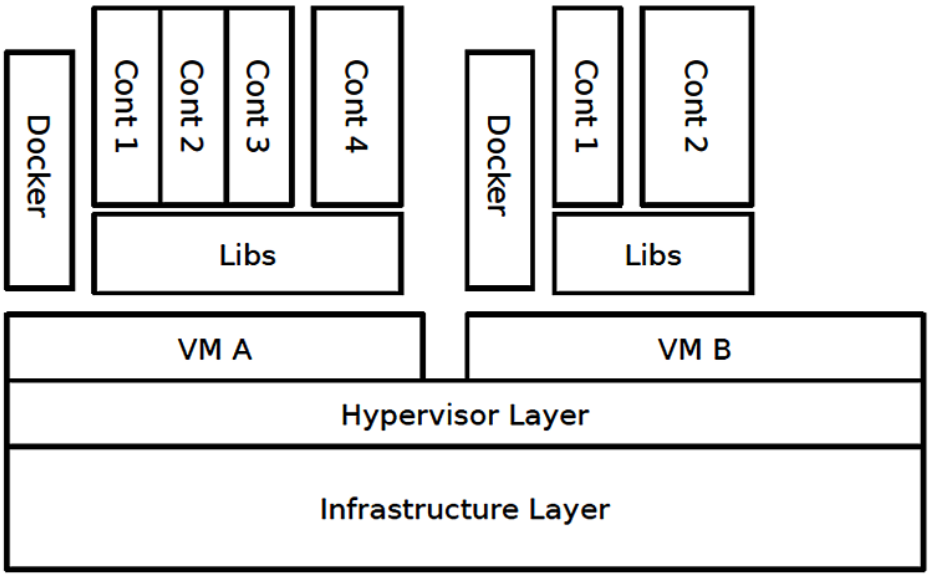
Similarly, Forestiero et al. (Forestiero, Mastroianni, Meo, Papuzzo, & Sheikhalishahi, 2014) proposed EcoMultiCloud, a hierarchical approach for workload management that offers an energy efficient VM placement in a multi-site data center. Their proposed architecture consists of two main layers. The upper layer is responsible for the assignment of workload (virtual machine requests) among remote sites and lower layer places virtual machines to hosts in each site. The proposed hierarchical approach achieves same energy efficiency as ECE (centralized solution), and offers more flexibility. This is because, as a hierarchical approach, it allows single data centers to select their internal VM placement algorithms.

VM Sizing

Virtualization technology provides the opportunity for applications to share the underlying hardware with secure isolation (Meng et al., 2010). Virtual machines configuration, in terms of the amount of resources (CPU, memory and I/O), are pre-defined by the cloud provider in most of the cloud service models. VM configuration is important for the resource allocation process where a host with enough resources need to be chosen to host the VM. Such VM placement process may ultimately affect the energy consumption of the data center.

Therefore, the efficiency of VM placement can be achieved by three different approaches. In the first approach, VMs are assigned to hosts according to their fix sizes and consolidated to less number of servers without change of configuration. This approach is discussed with details in the VM consolidation section (A. Beloglazov, 2010; Gmach et al., 2009)beloglazov_adaptive_2010 (Anton Beloglazov & Buyya, 2012) MorenoInference:2013 (Moreno, Yang, et al., 2013) . The second approach is tailoring

Figure 11. Hybrid virtual environment



virtual machine configuration to the workload, which can be achieved through characterization of the applications workload. These two approaches are considered as *Static VM Sizing* (Figure 10). Finally, the third approach is adjusting the VM's configuration to match its workload in runtime (Meng et al., 2010) and is known as *Dynamic VM Sizing* (As shown in Figure 10).

Static VM Sizing

Assuncao et al. (Assuncao et al., 2012) proposed *CloudAffinity*, a framework to match physical servers to VM instances called as *CloudMates*. This framework enables organizations to move their workloads to the cloud while choosing optimal number of available VM templates considering their budget constraint. *CloudAffinity* considers CPU, memory, and disk requirement of each server and chooses the optimal number of VM templates minimizing the user's cost based on the predefined Quality of Service (QoS). The QoS is defined as the percentage of the requests which are satisfied by each VM template. The effectiveness of the VM template matching is investigated through three metrics including cost, Euclidean distance, and Matching factor. The cost metric shows the amount of money that the user should pay to maintain a cloud instance and this cost differs from one instance to the other. The Euclidean distance metric is the distance between the cloud provider's template and the user's requirement in terms of the resources including CPU, memory and disk. The Matching factor metric shows the percentage of the difference between customer's requirement and what the template offers for each VM.

Piraghaj et al.(S.F. Piraghaj et al., 2015) also investigated the effect of virtual machine configurations on the total energy consumption of the data center. However, Piraghaj et al.(S.F. Piraghaj et al., 2015) tailored the VM configurations to the workload, instead of choosing from the available configurations which is the case of CloudAffinity (Assuncao et al., 2012).

Table 8. Energy efficient research considering hybrid virtual environment

Authors	Workload	SLA	Energy Saving	Energy Model
Dhyani et al. (Dhyani et al., 2010)	Synthetic Workload	NA	Efficient allocation of resources	NA
Bichler et al. (Bichler et al., 2006)	Traces from 30 dedicated servers hosting different types of application services	NA	Efficient resource allocation, Decreasing the number of required servers for hosting applications	NA
Tchana et al. (Tchana et al., 2015)	An enterprise Internet application benchmark (SPECjms2007 benchmark http://www.spec.org/jms2007/)	Service degradation threshold defined at start time for each application	Software Consolidation	NA
Yaqub et al. (Yaqub et al., 2014)	High variability datasets from Google are characterized for OpenShift cloud which can span multi-domain IaaS.	SLA violation (SLAV) is modeled as upper bound estimate considering performance degradation due to both migration (PDM) and contention on machine's resources (PDC)	Software Consolidation, Container migration	Linear Power Model (CPU only)
Almeida et al. (Almeida et al., 2006)	Transactional web services workload is classified into independent Web service (WS) classes.	Response time	Efficient resource allocation	NA

In order to have efficient configurations, VM sizes are tailored to the workload. In this respect, an analysis of the system workload is inevitable. The work is carried out in three major steps, firstly the usage patterns of tasks are studied and the similarities in these patterns initiated the idea of grouping the tasks according to their utilization in terms of CPU and memory. In the second step, the clustering output is used for identifying the VM configuration for each group of tasks separately and in the third step, each group of tasks is mapped to a corresponding VM size. The presented approach is validated through simulation and Google backend data is utilized as the input workload. The efficiency of the identified VM sizes through the proposed technique is further compared with some of the Amazon EC2 instances. It is also provided that workload characterization and the utilized feature-set play an important role on the efficiency of the identified VM sizes. In this respect, two different feature sets are selected for clustering the tasks. The first set contains the scheduling class[1], submission rate and the length of the task along with the average resource utilization including memory and CPU, and disk. While the second feature set only contains the average CPU and memory utilization of tasks during the studied period (1 day of the trace). The second clustering feature set is shown to identify more efficient VM sizes in terms of the required number of servers and the number of instantiated VMs.

Dynamic VM Sizing

In dynamic VM sizing, the approach estimates the amount of the resources that should be allocated to a VM with the objective of matching the VMs resources to its workload. The VM sizing should be carried out in a way that to avoid SLA violations resulted from under-provisioning of resources. Meng

et al. (Meng et al., 2010), present a provisioning approach in which the less correlated VMs are packed together and an estimate of their aggregate capacity requirements is predicted. This approach applies statistical multiplexing considering dynamic VM resource utilization and it makes sure that the utilization peaks of one VM do not necessarily coincide with the co-allocated VMs. Hence, the amount of resources allocated to each VM varies according to its workload.

Chen et al. (M. Chen et al., 2011) also investigated the problem of VM sizing in the provisioning process, so that more VMs can be hosted on servers. The estimated VM sizes are referred as effective size (*ES*) ("OpenVZ Virtuozzo Containers Wiki,"), which is determined through Statistical Multiplexing principles. These principles take into account the factors that might affect aggregated resource demand of the server on which the VM is placed. The effective size of a VM is affected by both its own demand and its co-allocated VMs considering the correlation coefficient. This effective sizing technique is demonstrated to save more energy than a generic consolidation algorithm.

Dynamic Voltage and Frequency Scaling (DVFS)

In addition to the *Bare Metal* environment, Dynamic Voltage Scaling has also been applied to virtualized environments. Laszewski et al. (von Laszewski et al., 2009) focused on the design and implementation of energy-efficient VM scheduling in a DVFS-enabled compute clusters. Jobs are assigned to preconfigured VMs and the VMs are shutdown when the jobs finish. The solution is proposed for high performance cluster computing, however since they consider virtualization technology, it can be implemented in a cloud environment as well. The scheduling algorithm operates considering two main approaches. It either optimizes the processor power dissipating by running the CPU at lower frequencies with the minimum effect on the overall performances of the VMs or schedules the VMs on CPUs with low voltages and tries not to scale up the CPU voltage.

Urgaonkar et al. (Urgaonkar, Kozat, Igarashi, & Neely, 2010) investigated the problem of optimal resource allocation and power efficiency in cloud data centers through online control decisions.

These decisions are made utilizing available queueing information in the system and the Lyapunov optimization theory. Heterogeneous applications with volatile workloads are used to show that the presented approach can handle unpredictable changes in the workload since it is not dependent on any prediction or estimate of the workload. In the studied system, applications execute inside virtual machines and each application can have multiple instances running across different VMs. Dynamic Voltage and Frequency scaling of CPU is applied for improving the energy consumption of the data center. The frequency of CPU is decided according to the workload by the *Resource Controller*, which is installed on each server.

Authors conclude that DVFS does not always result in more energy savings and operators should also consider utilizing the low power modes available in modern processors which might provide better energy savings with the least performance degradation.

Hybrid

The hybrid virtualization model shown in Figure 11 is a combination of system and OS-level virtualization approaches. The containerization technology used in this model is mainly application containers such as Docker (Merkel, 2014), which was explained previously. This new model is currently provided by Google Container Engine and Amazon ECS as a new cloud computing service called Container as a

Service. Running containers inside virtual machines provides an extra layer of isolation while ensuring the required security for users. Research in this area are summarized in Table 8.

As discussed, VM consolidation can be utilized for reducing energy consumption in data centers. However, VM consolidation is limited by the VMs memory (Tchana et al., 2015) and unlimited number of VMs can not be mapped on a physical machine (PM). Therefore, in order to save more energy, VM's resources should also be utilized efficiently. This problem is tackled in the hybrid model through consolidation of containers on VMs, which further improves the utilization of VMs resources. This approach is introduced by Tchana et al. (Tchana et al., 2015) and is called *Software/Service Consolidation* problem (*SCP*). In the *SCP* problem, several software/services are dynamically collocated on one VM. The objective is reducing the number of VMs and consequently decreasing the population of PMs along with the data center power consumption. In order to provide the required isolation, applications execute inside Docker containers. The problem is modeled utilizing Constraint Satisfaction Programming (CSP) and the *Software Consolidation* is carried out along with the VM consolidation. In order to accelerate the software consolidation process, the search domain is reduced considering a couple of boundaries such as containers collocation constraints. One of the limitations of this approach is the OS of the VM hosting the container, since unlike virtual machines, containers share the OS with their host.

Yaqub et al. (Yaqub et al., 2014) also investigated SCP in PaaS Clouds. This problem is framed leveraging the Google definition of Machine Reassignment model for the ROADEF/EURO challenge Roadef (2016) and was extended for RedHat's public PaaS (OpenShift (OpenShift, 2016)). Four Meta-heuristics are applied to find solutions for (re)allocations of containers. The solutions are then compared and ranked considering SLA violations, energy consumption, resource contention, migrations, machine used, and utilization metrics for four different cloud configurations. Contrary to (Tchana et al., 2015), no boundaries are considered to reduce the search domain for the Meta-heuristics to speed up the consolidation process.

Almeida et al. (Almeida, Almeida, Ardagna, Francalanci, & Trubian, 2006) investigated the SCP problem in a Service-Oriented Architecture. The problem was divided into two related sub-problems, short-term resource allocation and long-term capacity planning. The short-term resource allocation problem has a short-term impact on the revenue and its solution determines the optimal resource allocation to different services while increasing the revenue obtained through SLA contracts. However, the answer to long-term problem determines the optimal size of the service center that maximizes the long-term revenue from SLA contracts along with decreasing the Total Cost of Ownership (TCO). These problems are modelled in the proposed framework and a deep analysis of effects of short-term resource allocation is provided. A model is presented for identification of the optimal resource allocation in order to maximize the revenues of the service provider while meeting the required QoS. Resource utilization and the associated costs are also taken into account. The proposed optimal model is fast in terms of the computation speed, which makes it a good candidate for online resource management. Transactional Web services are considered as the hosted applications in the data center. The services are categorized into sub-classes because of the volatility of the web server workloads. Each VM is responsible for one class of web servers (WS). In order to insure the quality of service for each class of the WS, admission control is employed on top of each VM which decides to accept or reject the requests.

Dhyani et al. (Dhyani, Gualandi, & Cremonesi, 2010), introduced a constraint programming approach for the SCP problem. The research objective is decreasing data center cost through hosting multiple services running in VMs on each host. The SCP is modeled as an Integer Linear Programming (ILP) problem and compared with the presented solution through constraint programming. The constraint

programming approach can find the solution in less than 60 seconds. However, ILP could find a better solution if it could meet the 60 seconds deadline. Therefore, constraint programming is found as a better solution comparing to ILP for SCP problem considering the algorithm speed.

Bichler et al. (Bichler, Setzer, & Speitkamp, 2006) also investigated the problem of capacity planning. An IT service provider hosting services of multiple customers is investigated in a virtualized environment. Three capacity planning models are proposed for three allocation problems. These problems are solved through multi-dimensional bin-packing approximate algorithms and the workloads of 30 services are applied as the input of the system.

Piraghaj et al. (S. F. Piraghaj, Dastjerdi, Calheiros, & Buyya, 2015), investigated energy-efficient resource management algorithms for container as a Service (CaaS) cloud model. A framework is presented to tackle the energy efficiency issue in the context of CaaS through container consolidation. The CaaS environment is modeled and four sets of simulation experiments are carried out and their impact on system performance and data center energy consumption is evaluated. Four placement algorithms are utilized for identifying the destination host.

A container selection algorithm is responsible for selecting the containers to migrate from an overloaded host. Three selection algorithms including random, correlation aware, which selects the most correlated container with the host load, and the most utilized container in terms of CPU are compared. The algorithm which selects the most correlated container is identified the most efficient one in terms of the energy consumption.

WORKLOAD CHARACTERIZATION AND MODELING

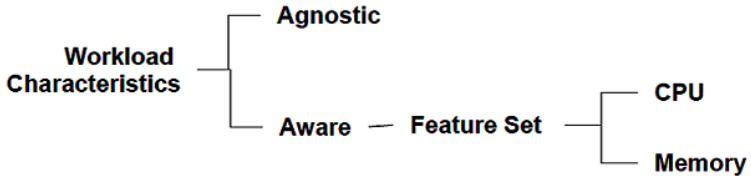
There is a growing body of research on resource management techniques with the focus on minimizing the energy usage in cloud data centers (Kansal, Zhao, Liu, Kothari, & Bhattacharya, 2010; Nathuji & Schwan, 2007). These techniques should be applicable for dynamic cloud workloads. However, because of the competitiveness and security issues, cloud providers do not disclose their workloads, and as a result there are not many publicly available cloud back-end traces. Therefore, most of the research lacks the study of the dynamicity in users demand and workload variation. The availability of cloud backend traces makes researchers able to model real cloud data center workloads. The obtained model can be applied for proving the applicability of the proposed heuristics in real world scenarios.

In 2009, Yahoo released traces from a production MapReduce M45 cluster to a selection of universities (Yahoo, 2010). In the same year, Google made the first version of its traces publicly available and this publicity resulted in a variety of research investigating the problems of capacity planning and scheduling via workload characterisation and statistical analysis of the planet's largest cloud backend traces (Reiss, Wilkes, & Hellerstein, 2011).

Workload Definition

The performance of a system is affected not only by its hardware and software components but also by the load it has to process (Calzarossa & Serazzi, 1993). As stated by Feitelson (Feitelson, 2015), understanding the workload is more important than designing new scheduling algorithms. If the tested systems do not have its input workload chosen correctly, the result of the proposed policies or algorithms might not work as expected when applied to real world scenarios.

Figure 12. The energy efficient resource management techniques in PaaS environment are grouped based on the approach awareness of the cloud workload and its characteristics



The computer workload is defined as the amount of work allocated to the system that should be completed in a given time. A typical system workload consists of tasks and group of users who are submitting the requests to the data center. For example, in Google workload tasks are the building block of a job. In other words, a typical job consists of one or more tasks (Reiss et al., 2011). These jobs are submitted by the users, which are in this case the Google's engineers or its services.

Workload Modeling Techniques

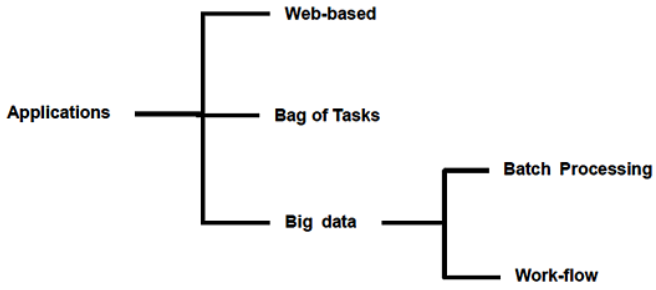
In order to characterize the workload, the drive or input workload of the studied system should also be investigated. For measuring the performance of a computer system, input workload[2], should be the same as the real one. As stated by Ferrari (Ferrari, 1972), there are three types of techniques for obtaining the input workload:

- **Natural Technique:** Natural technique utilizes real workloads obtained from the log file of the system without any manipulation. Urgaonkar et al. (Urgaonkar et al., 2010), utilized real traces from heterogeneous applications to investigate the problem of optimal resource allocation and power efficiency in cloud data centers. Anselmi et.al (Anselmi et al., 2008) also applied real workloads from 41 servers to validate their proposed approach for Service Consolidation Problem (SCP). PlanetLab VMs traces are applied as the input workload to validate the consolidation technique in several works (Anton Beloglazov & Buyya, 2012; Buyya et al., 2010; S. F. Piraghaj et al., 2015).
- **Artificial Technique:** Artificial technique involves the design and application of a workload that is independent of the real one. Mohan Raj and Shriran (Mohan Raj & Shriram, 2011) apply synthetic workloads following the Poisson distribution to model web server workloads.
- **Hybrid Technique:** Hybrid technique involves sampling a real workload and constructing the test workload from the parts of the real workload. Hindman et al. (Hindman et al., 2011) evaluate Mesos the application of both CPU and IO-intensive workloads that are derived from the statistics of Facebook cloud backend traces and running applications utilizing Hadoop and MPI.

Workload Modelling

As stated by Calzarossa and Swerazzi (Calzarossa & Serazzi, 1993), the workload modeling process can be constructed through three main steps. The first step is the formulation in which the basic components such as submission rates for users and their descriptions are selected. In addition to this, for evaluating

Figure 13. Application types supported in energy management systems



the proposed model, a criteria is considered. During the second step, the required parameters for modeling are collected while the workload executes in the system. Finally in the last step, a statistical analysis is performed on the collected data.

In selecting the workload modeling technique, the considered parameters for defining the requests play an important role (Agrawala, Mohr, & Bryant, 1976). In a distributed system, a user request is mainly defined via three main parameters including:

1. **t:** The time **t** is when the request is submitted to the system.
2. **l:** The location **l** is where the request is submitted from.
3. **r:** The request vector **r** contains the amount of resources needed in terms of CPU, memory and disk.

When time and spatial distribution of the user requests are ignored, e.g., only one day of the trace is studied, requests population are likely to have similarities and can be presented in the form of relatively homogeneous classes (Agrawala et al., 1976). Such kind of workload modeling is explored by Mishra et al. (Mishra, Hellerstein, Cirne, & Das, 2010) and Chen et al. (Y. Chen, Ganapathi, Griffith, & Katz, 2010) on the first version of the Google cluster traces. Mishra et al. (Mishra et al., 2010) applied the clustering algorithm K-means for forming the groups of tasks with more similarities in resource consumption and duration, while Chen et al. (Y. Chen et al., 2010) classified jobs instead of tasks. In addition to these approaches, Di et al. (Di, Kondo, & Cappello, 2013) characterized applications running in the Google cluster. Like (Y. Chen et al., 2010; Mishra et al., 2010), K-means is chosen for the clustering purpose. In our previous (Sareh Fotuhi Piraghaj, Calheiros, Chan, Dastjerdi, & Buyya, 2016), we proposed an end-to-end architecture aiming at efficient resource allocation and energy consumption in cloud data centers. In the presented architecture, the knowledge obtained from the analysis of the cloud backend workload is utilized to define customized virtual machine configuration along with maximum task capacity of each VM. Like the other aforementioned works (Moreno, Garraghan, Townend, & Xu, 2013; Solis Moreno, Garraghan, Townend, & Xu, 2014), the availability of virtualization technology is considered and the tasks are executed on top of virtual machines instead of physical servers. Unlike other approaches, the aim is decreasing energy by defining the virtual machines configurations along with their maximum task capacity.

If the time and location of the requests are considered, the workload can be modeled via a stochastic process such as Markovian model or time series models such as the technique applied by Khan et al. (Khan, Yan, Tao, & Anerousis, 2012). Khan et al. (Khan et al., 2012) presented an approach based on

Hidden Markov Modeling (HMM) to characterize the temporal correlations in the clusters of VMs that are discovered and to predict the patterns of workload along with the probable spikes.

Workload-Based Energy Saving Techniques

Study of the characteristics of the workload and its fluctuations is crucial for selecting energy management techniques. For example in Intel Enhanced Speed Stepping Technology (Intel, 2016), the CPU frequency and voltage are dynamically adjusted according to the servers workload. From the analysis of the workload, one can decide if a power management methodology is applicable for the system. As stated by Dhiman et al. (Dhiman et al., 2008), DVFS does not always result in more energy savings and operators should also consider utilizing low power modes available in modern processors that might provide better energy savings with the least performance degradation considering the workload. The workload type is also important for DVFS on memory component because, as stated previously, in non-memory intensive workloads running at lower memory speed would result in less performance degradation than memory-intensive workloads. Therefore, reducing power consumption can be obtained through running memory at a lower frequency with the least effect on the application performance (David et al., 2011). The energy efficient resource management techniques in PaaS environments are grouped into two major categories namely workload aware and workload agnostic as depicted in Figure 12.

Beloglazov et al. (Anton Beloglazov & Buyya, 2012) applied Markov chain model for known stationary workloads while utilizing a heuristic-based approach for unknown and non-stationary workloads. Apart from this work, the analysis of workloads of co-existing/co-allocated VMs motivated new algorithms and management techniques for saving energy in cloud data centers. These techniques contain the interference-aware (Caglar et al., 2013; Moreno, Yang, et al., 2013; Nathuji et al., 2010) and correlation-aware and multiplexing (M. Chen et al., 2011; Ferdman et al., 2012; Meng et al., 2010; Verma et al., 2009) VM placement algorithms, virtual machine static (Assuncao et al., 2012) and dynamic sizing techniques (Meng et al., 2010), which were discussed previously. The workload study also motivated the idea of overbooking resources to utilize the unused resources allocated to the VMs (Hu et al., 2013; Svärd et al., 2011; Tomas et al., 2014; Tomás & Tordsson, 2014).

APPLICATION-BASED ENERGY SAVING TECHNIQUES

The type of application (Figure 13) plays an important role in selecting the energy management technique. For scale out applications, turning on/off cores, which is called dynamic power gating, is not practical since these applications are latency sensitive and their resource demand is volatile, therefore the transition delay between power modes would degrade the QoS. In this respect, Kim et al. (Jungsoo Kim et al., 2013) considered the number of cores according to the workloads peak and achieved power efficiency through DVFS.

Web Applications

Web applications deployed in cloud data centers have highly fluctuating workloads. Wang et al. (D. Wang et al., 2013) measured the impact of utilizing DVFS for multi-tier applications. They concluded that response time and throughput are considerably affected as results of bottlenecks between the database

and application servers. The main challenge is identifying the DVFS adjustment period, which is not synchronized with workload burst cycles. Therefore, they proposed a workload-aware DVFS adjustment method that lessens the performance impact of DVFS when a cloud data center is highly utilized. VM consolidation methods also have been used along with DVFS for power optimization of multi-tier web applications.

Wang et al (L. Wang, Tao, von Laszewski, & Chen, 2010) proposed a performance-aware power optimization that combines either DVFS or VM consolidation. To achieve the maximum energy efficiency, they integrate feedback control with optimization strategies. The proposed approach operates in two levels: 1) at the application level, it uses a multi-input-multi-output controller to reach the performance stated in SLA by dynamically provisioning VMs, reallocating shared resources across VMs and DVFS, 2) at the data center level, it consolidates VMs onto the most energy-efficient host.

Bag of Tasks

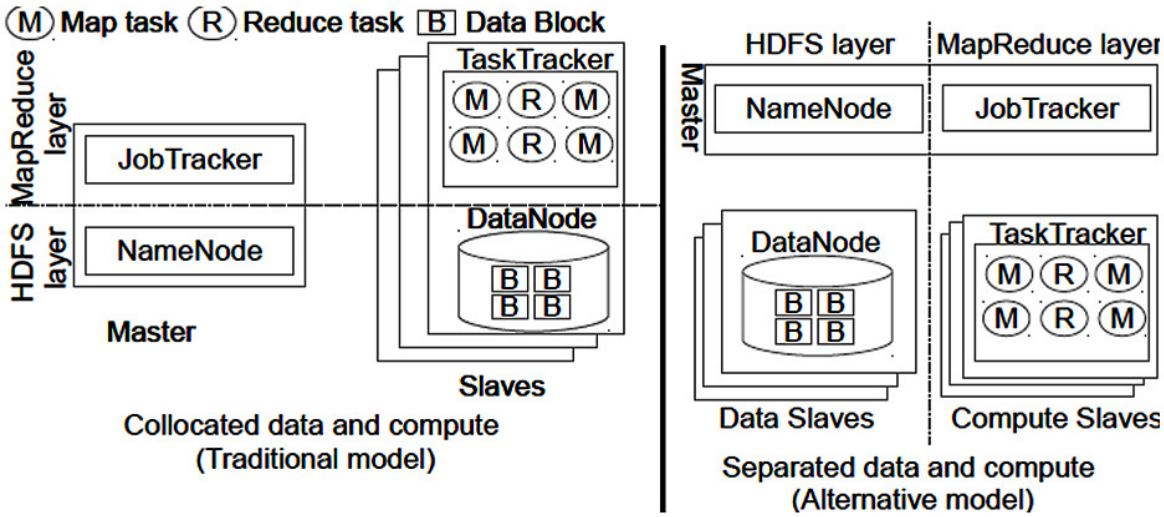
Bag-of-Tasks (BoT) applications are defined as parallel applications whose tasks are independent (Cirne et al., 2003). Kim et al. (K. H. Kim et al., 2007) investigated the problem of power-aware BoT scheduling on DVS-enabled cluster systems. Applying DVFS capability of processors, the presented space-shared and time-shared scheduling algorithms both saved a considerable energy while meeting the user-defined deadline.

Calheiros et al. (Calheiros & Buyya, 2014) proposed an algorithm for scheduling urgent, CPU intensive Bag of Tasks (BoT) utilizing processors DVFS with the objective of keeping the processor at the minimum frequency possible while meeting the user-defined deadline. An urgent application is defined as a High Performance Computing application that needs to be completed before the soft deadline defined by the user. Disaster management and healthcare applications are examples of this kind of applications. DVFS is applied at the middleware/Operating System level rather than at CPU level and maximum frequency levels are supplied by the algorithm during task execution. The approach does not require prior knowledge of the host workload for making decisions.

Big Data Applications

As indicated by International Data Corporation (IDC) in 2011, the overall information created and copied in the world has grown by nine times within five years reaching 1.8 zettabytes (1.8 trillion gigabytes) (Gantz & Reinsel, 2011) and this trend would continue to at least double every two year. The exceptional growth in the amount of produced data introduced the phenomenon named *Big Data*. There exist various definitions for Big Data. However, Apache Hadoop definition is the one which is close to the concept of this study. Apache Hadoop defines Big Data as "datasets that could not be captured, managed and processes by general computers within an acceptable scope". Big data analysis and processing along with the data storage and transmission require huge data centers that would eventually consume large amount of energy. In this respect, energy efficient power management techniques are really crucial for Big Data processing environments. In this section, we discuss batch processing and workflows as two examples of Big Data applications along with the techniques applied to make them more energy efficient (See Figure 13).

Figure 14. Two MapReduce development models studied in Feller, Ramakrishnan, and Morin
Source: Feller, Ramakrishnan & Morin, 2015



Batch Processing

Large-scale data analysis and batch processing are enabled utilizing data center resources through parallel and distributed processing frameworks such as MapReduce (Dean & Ghemawat, 2008) and Hadoop (Hadoop, 2016). The large scale data analysis performed by these frameworks requires many servers and this triggers the possibility of a considerable energy savings that can be obtained via resource management heuristics that minimize the required hardware.

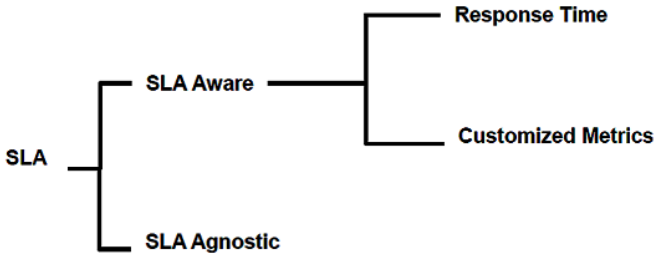
As stated by Leverich et al. (Leverich & Kozyrakis, 2010), MapReduce is widely used by various cloud providers such as Yahoo and Amazon. Google executes on average one hundred thousand

MapReduce jobs every day on its clusters("Google Open Source Blog: An update on container support on Google Cloud Platform,").The vast usage of this programming model, along with its unique characteristics, requires further study to explore any possibilities and techniques that can improve energy consumption in such environments.

The energy saving in a cluster can either be made by limiting the number of active servers to the workload requirement and shutting down the idle servers or matching the compute and storage of each server to its workloads. Due to the special characteristics of the MapReduce frameworks, these options are not useful in these environments. Powering down idle servers is not applicable, as in MapReduce frameworks data is distributed and stored on the nodes to ensure reliability and availability of data. Therefore, shutting down a node would affect the performance of the system and the data availability even if the node is idle. Moreover, in a MapReduce environment, the mismatch between hardware and workload characteristics might also result in energy wastage (e.g. CPU idleness for I/O workloads). Also, recovery mechanisms applied for hardware/software failures increases energy wastage in MapReduce frameworks.

Leverich et al. (Leverich & Kozyrakis, 2010) investigated the problem of energy consumption in Hadoop as a MapReduce style framework. Two improvements are applied to the Hadoop framework. Firstly, an energy controller is added that can communicate with the Hadoop framework. The second

Figure 15. The energy efficient resource management techniques for PaaS environments are categorized in two groups, namely SLA Aware and SLA Agnostic considering SLA



improvement is in the Hadoop data-layout and task distribution to enable more nodes to be switched off. The data-layout is modified so that at least one replica of a data block would be placed on a set of nodes referred as *Covering Set* (CS). These *Covering Sets* ensure the availability of the data block when the other nodes that store the other replicas are all shutdown to save power. The number of replicas in a Hadoop framework is specified by users and is equal to three by default.

Lang et al. (Lang & Patel, 2010) proposed a solution called All-In Strategy (AIS) that utilizes the whole cluster for executing the workload and then power down all the nodes. Results show that the effectiveness of the algorithms directly depend on both complexity of the workloads and the time it takes for the nodes to change power states.

Kaushik et al. (Kaushik, Bhandarkar, & Nahrstedt, 2010), presented GreenHDFS, an energy efficient and a highly scalable variant of the Hadoop Distribution File System (HDFS). GreenHDFS is based on the idea of energy-efficient data-placement through dividing servers into two major groups namely Hot and Cold zones. Data that are not accessed regularly are placed in the Cold zone so that a considerable amount of energy can be saved harnessing the idleness in this zone.

Long predictable, streaming I/O and parallelization and non-interactive performance are named as the characteristics of MapReduce workloads computations in Leverich et al. (Leverich & Kozyrakis, 2010). However, there exists MapReduce with interactive analysis (MIA) style workloads that have been widely used by organizations (Y. Chen, Alspaugh, Borthakur, & Katz, 2012). Since MapReduce makes storing and processing of large scale data a lot easier, data analysts are widely adopting MapReduce to process their data.

Typical energy saving solution obtained through maximization of server utilization is not applicable for MIA workloads because of two main reasons. Firstly, MIA workloads are dominated by human-initiated jobs that force the cluster to be configured to the peak load so that it can satisfy SLAs. Secondly, workload spikes are unpredictable and the environment is volatile because machines are added or removed from the cluster regularly. In this respect, Chen et al. (Y. Chen et al., 2012) proposed BEEMR (Berkeley Energy Efficient MapReduce) as an energy efficient MapReduce workload manager inspired by an analysis of the Facebook Hadoop workload.

Hadoop is the open source implementation of the MapReduce programming model. Apart from energy consumption, which is studied in a number of works (Y. Chen et al., 2012; Kaushik et al., 2010; Lang & Patel, 2010; Leverich & Kozyrakis, 2010) Hadoop performance for both collocated and separated compute services and data models (Figure 14) is investigated by Feller et al. (Feller et al., 2015). The separation of compute services and data is applied for virtualized environments. It is shown that the col-

location of VMs on servers has a negative effect on the I/O throughput, which makes physical clusters more efficient in terms of the performance when compared to the virtualized clusters. The performance degradation is proven to be application-dependent and related to the data-to-compute ratio. There is also a trade-off between the application's completion time and the energy consumed in the cluster.

Workflow Applications

Workflows or precedence-constrained parallel applications are a popular paradigm for modeling large applications that is widely used by scientists and engineers. Therefore, there has been an increasing effort to improve the performance of these applications through utilizing distributed resources of Clouds. With the increase in the interest toward this type of applications, the energy efficiency of the proposed approaches also comes into the picture, as performance efficiency brought by excessive use of resources might result in extra energy consumption.

The inefficiency of provisioned resources for scientific workflows execution results in excessive energy consumption. Lee et al. (Lee, Han, Zomaya, & Yousif, 2015) addressed this issue through a resource-efficient workflow scheduling algorithm named MER. The proposed algorithm optimizes the resource usage of a workflow schedule generated by other scheduling algorithms. MER consolidates tasks that were previously scheduled and maximizes the resource utilization. Based on the trade-off between makespan (execution time) increase and resource utilization reduction, MER identifies the near optimal trade-off point between these two factors. Finding this point, the algorithm improves resource utilization and consequently reduces the provisioned resources and saves energy. The proposed algorithm can be applied to any environment in which scientific workflows of many precedence-constrained tasks are executed. However, MER is specifically designed for the IaaS cloud model.

As discussed earlier, Dynamic Voltage and Frequency Scaling (DVFS) is an effective approach to minimize the energy consumption of applications. As scientific workflows contain tasks with data dependencies between them, DVFS might not always result in desirable energy saving. Depending on system and workflow characteristics, decreasing the CPU frequency may increase the overall execution time and the idle time of the processors, which consequently deteriorates the planned energy saving. In addition, when the SLA violation penalty is higher than the power savings, adjusting the CPU to operate at the lowest frequency is not always energy efficient. In this situation, executing the tasks quickly with a higher frequency might result in less energy consumption (Freeh et al., 2007). In this respect, Pietri et al. (Pietri & Sakellariou, 2014) proposed an algorithm that identifies the best time to reduce the frequency in a way that the overall energy consumption is decreased. In the presented approach, the lowest possible frequency did not always result in the least energy consumption for completing the workflow execution. The algorithm considers various task runtime and processor frequency capabilities and it assumes an initial task placement on the available machines. Next, it determines the appropriate CPU frequency considering the time that the task can be stretched without violating the deadline (slack time). The proposed algorithm gradually scales down the frequency of the processor assigned for each task iteratively by the time the overall energy savings are increased. In each iteration, the CPU frequency is scaled down to the next available frequency mode. The algorithm performance is validated through simulation and the results demonstrated that the system can provide a good balance between energy consumption and makespan.

Durillo et al. (Durillo, Nae, & Prodan, 2014) proposed MOHEFT as an extension of the Heterogeneous Earliest Finish Time (HEFT) algorithm (Topcuoglu, Hariri, & Wu, 2002), which is widely applied for workflow scheduling. The proposed algorithm is able to compute a set of suboptimal solutions in a single run without any prior knowledge of the execution time of tasks. MOHEFT policy complements the HEFT scheduling algorithm through predicting task execution time based on the historical data obtained from real workflow task executions.

SLA AND ENERGY MANAGEMENT TECHNIQUES

The expectations of providers and costumers of a cloud service including the penalties considered for violations are all documented in the Service Level Agreement SLA) (Greenwood, Vitaglione, Keller, & Calisti, 2006; Hani, Paputungan, & Hassan, 2015; Yogamangalam & Sriram, 2013). Considering SLA, energy management techniques are categorized into two groups, namely SLA-Aware and SLA-Agnostic approaches (as shown in Figure 15).

SLA contains service level objectives (SLOs) including the service availability and performance in terms of the response time and throughput (W. Kim, 2013). Satisfying SLA in a cloud computing environment is one of the key factors that builds trust between consumers and providers. There has always been a trade-off between saving energy and meeting SLA in resource management policies, therefore it is really crucial to make sure that energy saving does not increase SLA violations dramatically.

The metrics utilized to measure SLA can be different based on the application type, for example SLA for workflow applications is defined in terms of the user-defined deadlines (Durillo et al., 2014; Lee et al., 2015; Pietri & Sakellariou, 2014) while in web and scale-out applications it is defined as the response time (Anselmi et al., 2008; Hindman et al., 2011; Lang & Patel, 2010). Anselmi et.al (Anselmi et al., 2008) consider the application response time as their SLA metric in their proposed solution for the Service Consolidation Problem (SCP) considering multi-tier applications.

In the studied scenario, the objective was minimizing the number of required servers while satisfying the Quality of Service. Similarly, Mohan et al. (Mohan Raj & Shriram, 2011) considered response time of the application as the SLA metric in their proposed energy efficient workload scheduling algorithm. The application request is accepted considering the data center capacity along with the SLA. The SLA is maintained through a control theoretic method. Holt-Winters forecasting formula is applied for improving the SLA through minimizing the incurred cost by the time in which the system waits for startup and shutdown delays of a PM/VM. Caglar et al. (Caglar et al., 2013) also considered response time as their SLA metric in the presented online VM placement technique. In a different approach, Beloglazov et al. (A. Beloglazov, 2010) utilized a combined metric considering both SLA violation and energy consumption for their optimization problem. In the presented approach, SLA is violated during the time that a host is overloaded. This approach is application independent.

SUMMARY

According to the Refrigerating and Air Conditioning Engineers (ASHRAE) (Belady & Beaty, 2005), the Infrastructure and Energy Cost (I&E) has increased by 75% of the cost in 2014 while IT costs are only 25%(“In the data center, power and cooling costs more than the it equipment it supports « Electronics

Cooling Magazine – Focused on Thermal Management, TIMs, Fans, Heat Sinks, CFD Software, LEDs/ Lighting,"). This is a significant rise for I&E costs which was contributing up to 20% to the whole cost when IT costs where only 80% in the early 90's. This drastic rise of data center power consumption has made energy management techniques a non-separable part of the resource management in a cloud computing environment. In this respect, there is a large body of literature that consider energy management techniques for various cloud service models. In this chapter, we mainly focused on the PaaS service model in which the data center owner can obtain prior knowledge of the applications and their usage patterns. Further, we discuss the energy management techniques in both bare-metal and virtualized environments. In summary, research in this area concludes that selecting the right energy management technique is dependent on three main factors:

- **The Environment Where the Applications Run:** In this chapter, we covered various alternatives for execution environments including Bare Metal, containerized, and hypervisor-based virtualization.
- **The Workload and Application Type:** Applications are mainly different in terms of their workload patterns, latency sensitiveness, and etc. Understanding the workload characteristics can further improve the efficiency of the algorithms.
- **The Quality of Service:** The QoS for applications is defined through the Service Level Agreements (SLA) and the SLA metric can be different considering the applications nature. Considering SLA is important since energy management techniques might result in SLA violations and consequently degrade the performance of the system or increase the total costs for the applications execution.

Considering these factors, for future studies on this area, the following directions can further be developed:

- **Containerized Environment:** Further studies can be done on the consolidation of containers considering both the application and OS containers. The container migration capability can be studied deeply and might substitute VM migrations because of their smaller overhead and shorter startup delays.
- **Network-Aware VM/Container Consolidation:** The network element can be considered as one of the factors in the consolidation to decrease the communication overhead between VMs/ containers.
- **CaaS Environment:** The newly introduced cloud service model introduces new research directions that are required to explore with more details. The research directions in this area include joint VM-container consolidation algorithms along with new SLA metrics.

REFERENCES

Agrawala, A. K., Mohr, J., & Bryant, R. (1976). An approach to the workload characterization problem. *Computer*, *9*(6), 18–32. doi:10.1109/C-M.1976.218610

Almeida, J., Almeida, V., Ardagna, D., Francalanci, C., & Trubian, M. (2006). Resource Management in the Autonomic Service-Oriented Architecture*Proceedings of the 2006 IEEE International Conference on Autonomic Computing (ICAC 2006)* (pp. 84-92). doi:10.1109/ICAC.2006.1662385

AMD. (2016). *AMD Turbo Core Technology*. Retrieved 31 March, 2016, from http://www.amd.com/en-us/innovations/software-technologies/turbo-core

Anselmi, J., Amaldi, E., & Cremonesi, P. (2008). Service Consolidation with End-to-End Response Time Constraints*Proceedings of 34th Euromicro Conference on Software Engineering and Advanced Applications (SEAA 2008.)* (pp. 345-352). doi:10.1109/SEAA.2008.31

Assuncao, M. D., Netto, M. A. S., Peterson, B., Renganarayana, L., Rofrano, J., Ward, C., & Young, C. (2012). CloudAffinity: A framework for matching servers to cloudmates*Proceedings of the 2012 IEEE Network Operations and Management Symposium (NOMS 2012)* (pp. 213-220). doi:10.1109/NOMS.2012.6211901

AWS. (2016). *EC2 Instance Types – Amazon Web Services (AWS)*. Retrieved 31 March, 2016, from https://aws.amazon.com/ec2/instance-types/

Barroso, L. A., & Holzle, U. (2007). The Case for Energy-Proportional Computing. *Computer*, *40*(12), 33–37. doi:10.1109/MC.2007.443

Belady, C. L., & Beaty, D. (2005). Roadmap for Datacom Cooling. *ASHRAE Journal*, *47*(12), 52.

Beloglazov, A. (2010). *Energy Efficient Allocation of Virtual Machines in Cloud Data Centers*. Paper presented at the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid 2010). doi:10.1109/CCGRID.2010.45

Beloglazov, A., & Buyya, R. (2010). *Adaptive Threshold-based Approach for Energy-efficient Consolidation of Virtual Machines in Cloud Data Centers*. Paper presented at the 8th International Workshop on Middleware for Grids, Clouds and e-Science, Bangalore, India. doi:10.1145/1890799.1890803

Beloglazov, A., & Buyya, R. (2010). Energy efficient allocation of virtual machines in cloud data centers*Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid 2010)* (pp. 577 - 578). doi:10.1109/CCGRID.2010.45

Beloglazov, A., & Buyya, R. (2012). Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers. *Concurrent Computing: Practice and Experience*, *24*(13), 1397–1420. doi:10.1002/cpe.1867

Bichler, M., Setzer, T., & Speitkamp, B. (2006). Capacity planning for virtualized servers*Proceedings of the 16th Annual Workshop on Information Technologies and Systems (WITS 2006)*.

Bottomley, J. (2013, May). *Containers and The Cloud A Match Made in Heaven*. Academic Press.

Brewer, E. (2016). *Google Open Source Blog: An update on container support on Google Cloud Platform*. Retrieved from http://google-opensource.blogspot.com.au/2014/06/an-update-on-container-support-on.html

Buyya, R., Beloglazov, A., & Abawajy, J. (2010). Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges*Proceedings of 16th International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010)* (pp. 6-17).

Caglar, F., Shekhar, S., & Gokhale, A. (2013). *A Performance Interference-aware Virtual Machine Placement Strategy for Supporting Soft Realtime Applications in the Cloud.* Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA, Tech. Rep. ISIS-13-105.

Calheiros, R. N., & Buyya, R. (2014). Energy-Efficient Scheduling of Urgent Bag-of-Tasks Applications in Clouds through DVFS*Proceedings of the 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2014)* (pp. 342-349). doi:10.1109/CloudCom.2014.20

Calzarossa, M., & Serazzi, G. (1993). Workload characterization: A survey. *Proceedings of the IEEE*, *81*(8), 1136–1150. doi:10.1109/5.236191

Charles, J., Jassi, P., Ananth, N. S., Sadat, A., & Fedorova, A. (2009). Evaluation of the Intel\textregistered Core™ i7 Turbo Boost feature*Proceedings of the 2009 IEEE International Symposium on Workload Characterization (IISWC 2009)* (pp. 188-197). doi:10.1109/IISWC.2009.5306782

Chen, M., Zhang, H., Su, Y.-Y., Wang, X., Jiang, G., & Yoshihira, K. (2011). Effective VM sizing in virtualized data centers*Proceedings of the 2011 IFIP/IEEE International Symposium on Integrated Network Management (IM 2011)* (pp. 594-601). doi:10.1109/INM.2011.5990564

Chen, Y., Alspaugh, S., Borthakur, D., & Katz, R. (2012). Energy Efficiency for Large-scale MapReduce Workloads with Significant Interactive Analysis*Proceedings of the 7th ACM European Conference on Computer Systems* (pp. 43-56). doi:10.1145/2168836.2168842

Chen, Y., Ganapathi, A. S., Griffith, R., & Katz, R. H. (2010). *Analysis and lessons from a publicly available Google cluster trace.* EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-95, University of California, Berkeley.

Cirne, W., Brasileiro, F., Sauvé, J., Andrade, N., Paranhos, D., Santos-neto, E., & Gr, F. C. et al. (2003). Grid computing for of Bag-of-Tasks applications*Proceedings of the 3rd IFIP Conference on E-Commerce, E-Business and EGovernment*.

Container, L. (2016). *Linux Containers*. Retrieved 31 March, 2016, from https://linuxcontainers.org/

David, H., Fallin, C., Gorbatov, E., Hanebutte, U. R., & Mutlu, O. (2011). Memory Power Management via Dynamic Voltage/Frequency Scaling*Proceedings of the 8th ACM International Conference on Autonomic Computing* (pp. 31-40). doi:10.1145/1998582.1998590

Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified Data Processing on Large Clusters. *Magazine Communications of the ACM*, *51*(1), 107–113. doi:10.1145/1327452.1327492

Deng, Q., Meisner, D., Bhattacharjee, A., Wenisch, T. F., & Bianchini, R. (2012). CoScale: Coordinating CPU and Memory System DVFS in Server Systems*Proceedings of the 45th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2012)* (pp. 143-154). doi:10.1109/MICRO.2012.22

Deng, Q., Meisner, D., Ramos, L., Wenisch, T. F., & Bianchini, R. (2011). MemScale: Active Low-power Modes for Main Memory*Proceedings of the 16th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2011)* (pp. 225-238). doi:10.1145/1950365.1950392

Dhiman, G., Pusukuri, K. K., & Rosing, T. (2008). Analysis of Dynamic Voltage Scaling for System Level Energy Management*Proceedings of the 2008 Conference on Power Aware Computing and Systems (HotPower 2008)* (pp. 9-9).

Dhyani, K., Gualandi, S., & Cremonesi, P. (2010). A Constraint Programming Approach for the Service Consolidation Problem. Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, (pp. 97-101).

Di, S., Kondo, D., & Cappello, F. (2013). Characterizing Cloud Applications on a Google Data Center-*Proceedings of the 42nd International Conference on Parallel Processing (ICPP 2013)* (pp. 468-473). doi:10.1109/ICPP.2013.56

Dong, Z., Zhuang, W., & Rojas-Cessa, R. (2014). Energy-aware scheduling schemes for cloud data centers on Google trace data*Proceedings of the 2014 IEEE Online Conference on Green Communications (OnlineGreencomm 2014)* (pp. 1-6). doi:10.1109/OnlineGreenCom.2014.7114422

Durillo, J. J., Nae, V., & Prodan, R. (2014). Multi-objective energy-efficient workflow scheduling using list-based heuristics. *Future Generation Computer Systems*, *36*, 221–236. doi:10.1016/j.future.2013.07.005

Feitelson, D. G. (2015). *Workload modeling for computer systems performance evaluation*. Cambridge University Press. doi:10.1017/CBO9781139939690

Feller, E., Ramakrishnan, L., & Morin, C. (2015). Performance and energy efficiency of big data applications in cloud environments: A hadoop case study. *Journal of Parallel and Distributed Computing*, *79*, 80–89. doi:10.1016/j.jpdc.2015.01.001

Ferdman, M., Adileh, A., Kocberber, O., Volos, S., Alisafaee, M., Jevdjic, D., & Falsafi, B. et al. (2012). Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware. *ACM SIGPLAN Notices*, *47*(4), 37–48.

Ferrari, D. (1972). Workload characterization and selection in computer performance measurement. *Computer*, *5*(4), 18–24. doi:10.1109/C-M.1972.216939

Forestiero, A., Mastroianni, C., Meo, M., Papuzzo, G., & Sheikhalishahi, M. (2014). Hierarchical Approach for Green Workload Management in Distributed Data Centers*Proceedings of the 20th International European Conference on Parallel and Distributed Computing (Euro-Par 2014) Workshops* (pp. 323-334). doi:10.1007/978-3-319-14325-5_28

Forouzan, B. A. (2002). *TCP/IP protocol suite*. McGraw-Hill, Inc.

Freeh, V. W., Lowenthal, D. K., Pan, F., Kappiah, N., Springer, R., Rountree, B. L., & Femal, M. E. (2007). Analyzing the Energy-Time Trade-Off in High-Performance Computing Applications. *IEEE Transactions on Parallel and Distributed Systems*, *18*(6), 835–848. doi:10.1109/TPDS.2007.1026

Gantz, J., & Reinsel, D. (2011, June). *Extracting Value from Chaos*. Retrieved 10 May, 2016, from https://www.emc.com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf

Gmach, D., Rolia, J., Cherkasova, L., & Kemper, A. (2009). Resource Pool Management: Reactive Versus Proactive or Let's Be Friends. *Computer Networks*, *53*(17), 2905–2922. doi:10.1016/j.comnet.2009.08.011

Goldberg, R. P. (1974). Survey of Virtual Machine Research. *Computer*, *7*(9), 34–45. doi:10.1109/MC.1974.6323581

Graziano, C. D. (2011). *A performance analysis of Xen and KVM hypervisors for hosting the Xen Worlds Project*. Academic Press.

Greenberg, A., Hamilton, J., Maltz, D. A., & Patel, P. (2008). The cost of a cloud: Research problems in data center networks. *Computer Communication Review*, *39*(1), 68–73. doi:10.1145/1496091.1496103

Greenberg, S., Mills, E., Tschudi, B., Rumsey, P., & Myatt, B. (2006). Best practices for data centers: Lessons learned from benchmarking 22 data centers*Proceedings of the ACEEE Summer Study on Energy Efficiency in Buildings* (pp. 76-87).

Greenwood, D., Vitaglione, G., Keller, L., & Calisti, M. (2006). Service Level Agreement Management with Adaptive Coordination*Proceedings of the 2006 International conference on Networking and Services* (pp. 45-45). doi:10.1109/ICNS.2006.99

Hadoop. (2016). *Welcome to Apache™ Hadoop®!* Retrieved 10 May, 2016, from http://hadoop.apache.org/

Hani, A. F. M., Paputungan, I. V., & Hassan, M. F. (2015). Renegotiation in Service Level Agreement Management for a Cloud-Based System. *ACM Computer Survey, 47*(3), 51:51-51:21.

Haskell, B. G., Puri, A., & Netravali, A. N. (1997). *Digital Video: An Introduction to MPEG-2*. Springer Science & Business Media.

He, Y., Ye, Z., Fu, Q., & Elnikety, S. (2012). *Budget-based control for interactive services with adaptive execution*. Paper presented at the 9th International Conference on Autonomic Computing. doi:10.1145/2371536.2371557

Hecht-Nielsen, R. (1992). *Theory of the Backpropagation Neural Network* (Vol. 2). Neural Networks for Perception.

Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., & McKeown, N. (2010). ElasticTree: Saving Energy in Data Center Networks*Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation* (pp. 249-264).

Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A., Joseph, A. D., Katz, R., & Stoica, I. et al. (2011). Mesos: A Platform for Fine-grained Resource Sharing in the Data Center*Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation* (pp. 295-308).

Hoelzle, U., & Barroso, L. A. (2009). *The Datacenter As a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan and Claypool Publishers.

Hu, W., Hicks, A., Zhang, L., Dow, E. M., Soni, V., Jiang, H., & Matthews, J. N. et al. (2013). A quantitative study of virtual machine live migration*Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference* (pp. 11). doi:10.1145/2494621.2494622

Intel. (2016). *Enhanced Intel SpeedStep Technology (EIST)*. Retrieved 31 March, 2016, from http://www.intel.com/cd/channel/reseller/asmo-na/eng/203838.htmoverview

Kabir, M. H., Shoja, G. C., & Ganti, S. (2014). VM Placement Algorithms for Hierarchical Cloud Infrastructure*Proceedings of the 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2014)* (pp. 656-659). doi:10.1109/CloudCom.2014.53

Kansal, A., Zhao, F., Liu, J., Kothari, N., & Bhattacharya, A. A. (2010). Virtual Machine Power Metering and Provisioning*Proceedings of the 1st ACM Symposium on Cloud Computing (SoCC 2010)* (pp. 39-50). doi:10.1145/1807128.1807136

Kaplan, J. M., Forrest, W., & Kindler, N. (2008). *Revolutionizing data center energy efficiency*. Academic Press.

Kaufman, L., & Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis* (Vol. 344). John Wiley & Sons.

Kaushik, R. T., Bhandarkar, M., & Nahrstedt, K. (2010). Evaluation and Analysis of GreenHDFS: A Self-Adaptive, Energy-Conserving Variant of the Hadoop Distributed File System*Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science* (pp. 274-287). doi:10.1109/CloudCom.2010.109

Khan, A., Yan, X., Tao, S., & Anerousis, N. (2012). Workload characterization and prediction in the cloud: A multiple time series approach*Proceedings of the 2012 IEEE Network Operations and Management Symposium (NOMS 2012)* (pp. 1287-1294). doi:10.1109/NOMS.2012.6212065

Khosravi, A., Garg, S. K., & Buyya, R. (2013). Energy and carbon-efficient placement of virtual machines in distributed cloud data centers*Proceedings of the 19th International European Conference on Parallel and Distributed Computing (Euro-Par 2013)* (pp. 317-328). doi:10.1007/978-3-642-40047-6_33

Kim, J., Elnikety, S., He, Y., Hwang, S.-w., & Ren, S. (2013). *QACO: exploiting partial execution in web servers*. Paper presented at the 2013 ACM Cloud and Autonomic Computing Conference (CAC 2013). doi:10.1145/2494621.2494636

Kim, J., Ruggiero, M., Atienza, D., & Lederberger, M. (2013). Correlation-aware Virtual Machine Allocation for Energy-efficient Datacenters*Proceedings of the Conference on Design, Automation and Test in Europe* (pp. 1345-1350). doi:10.7873/DATE.2013.277

Kim, K. H., Buyya, R., & Kim, J. (2007). Power Aware Scheduling of Bag-of-Tasks Applications with Deadline Constraints on DVS-enabled Clusters*Proceedings of the 2007 IEEE Seventh International Symposium on Cluster Computing and the Grid (CCGrid 2007)* (pp. 541-548). doi:10.1109/CCGRID.2007.85

Kim, W. (2013). Cloud computing architecture. *International Journal of Web and Grid Services*, *9*(3), 287–303. doi:10.1504/IJWGS.2013.055724

Klein, C., & Maggio, M.-A. (2014). Brownout: building more robust cloud applications*Proceedings of the 36th International Conference on Software Engineering* (pp. 700-711).

Krioukov, A., Mohan, P., Alspaugh, S., Keys, L., Culler, D., & Katz, R. H. (2010). NapSAC: Design and Implementation of a Power-proportional Web Cluster*Proceedings of the First ACM SIGCOMM Workshop on Green Networking (Green Networking 2010)* (pp. 15-22). New York: ACM. doi:10.1145/1851290.1851294

Lang, W., & Patel, J. M. (2010). Energy Management for MapReduce Clusters. *Proceedings of the Very Large Data Bases Endowment Journal, 3*, 129-139. doi:10.14778/1920841.1920862

Lee, Y. C., Han, H., Zomaya, A. Y., & Yousif, M. (2015). Resource-efficient Workflow Scheduling in Clouds. *Knowledge-Based Systems*, *80*, 153–162. doi:10.1016/j.knosys.2015.02.012

Leverich, J., & Kozyrakis, C. (2010). On the Energy (in)Efficiency of Hadoop Clusters. *SIGOPS Operating Systems Review*, *44*(1), 61–65. doi:10.1145/1740390.1740405

Meisner, D., Gold, B. T., & Wenisch, T. F. (2009). PowerNap: Eliminating Server Idle Power. *SIGARCH Computer Architecture News*, *37*(1), 205–216. doi:10.1145/2528521.1508269

Meng, X., Isci, C., Kephart, J., Zhang, L., Bouillet, E., & Pendarakis, D. (2010). Efficient Resource Provisioning in Compute Clouds via VM Multiplexing*Proceedings of the 7th International Conference on Autonomic Computing* (pp. 11-20). doi:10.1145/1809049.1809052

Merkel, D. (2014). Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal, 2014*(239).

Mishra, A. K., Hellerstein, J. L., Cirne, W., & Das, C. R. (2010). Towards characterizing cloud backend workloads: Insights from Google compute clusters. *Performance Evaluation Review*, *37*(4), 34–41. doi:10.1145/1773394.1773400

Mohan Raj, V. K., & Shriram, R. (2011). Power aware provisioning in cloud computing environment*Proceedings of the 2011 International Conference on Computer, Communication and Electrical Technology (ICCCET)* (pp. 6-11). doi:10.1109/ICCCET.2011.5762447

Moreno, I. S., Garraghan, P., Townend, P., & Xu, J. (2013). An approach for characterizing workloads in Google cloud to derive realistic resource utilization models*Proceedings of the 7th IEEE International Symposium on Service Oriented System Engineering (SOSE 2013)* (pp. 49-60). doi:10.1109/SOSE.2013.24

Moreno, I. S., Yang, R., Xu, J., & Wo, T. (2013). Improved energy-efficiency in cloud datacenters with interference-aware virtual machine placement. *Autonomous Decentralized Systems (ISADS), 2013 IEEE Eleventh International Symposium on* (pp. 1-8). IEEE.

Nagy, G. (2015, May). *Operating System Containers vs. Application Containers*. Academic Press.

Nathuji, R., Kansal, A., & Ghaffarkhah, A. (2010). Q-clouds: managing performance interference effects for qos-aware clouds*Proceedings of the 5th European conference on Computer systems* (pp. 237-250). doi:10.1145/1755913.1755938

Nathuji, R., & Schwan, K. (2007). VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems*Proceedings of 21st ACM SIGOPS Symposium on Operating Systems Principles (SOSP 2007)* (pp. 265-278). doi:10.1145/1294261.1294287

Open, V. Z. (2016). *OpenVZ Virtuozzo Containers Wiki*. Retrieved 31 March, 2016, from https://openvz.org/Main_Page

OpenShift. (2016). *OpenShift QuickStart · GitHub*. Retrieved from https://github.com/openshift-quickstart

Pandit, D., Chattopadhyay, S., Chattopadhyay, M., & Chaki, N. (2014). Resource allocation in cloud using simulated annealing.*Proceedings of the 2014 Conference on Applications and Innovations in Mobile Computing (AIMoC 2014)* (pp. 21-27). doi:10.1109/AIMOC.2014.6785514

Pietri, I., & Sakellariou, R. (2014). Energy-Aware Workflow Scheduling Using Frequency Scaling. *Proceedings of the 43rd International Conference on Parallel Processing Workshops (ICCPW 2014)* (pp. 104-113). doi:10.1109/ICPPW.2014.26

Piraghaj, S. F., Calheiros, R. N., Chan, J., Dastjerdi, A. V., & Buyya, R. (2016). Virtual Machine Customization and Task Mapping Architecture for Efficient Allocation of Cloud Data Center Resources. *The Computer Journal*, *59*(2), 208–224. doi:10.1093/comjnl/bxv106

Piraghaj, S. F., Dastjerdi, A. V., Calheiros, R. N., & Buyya, R. (2015). A Framework and Algorithm for Energy Efficient Container Consolidation in Cloud Data Centers.*2015 IEEE International Conference on Data Science and Data Intensive Systems* (pp. 368-375). doi:10.1109/DSDIS.2015.67

Piraghaj, S. F., Dastjerdi, A. V., Calheiros, R. N., Buyya, R., Piraghaj, S. F., Dastjerdi, A. V., . . . Buyya, R. (2015). Efficient Virtual Machine Sizing for Hosting Containers as a Service (SERVICES 2015). *Proceedings of the 2015 IEEE World Congress on Services* (pp. 31-38).

PlanetLab. (2016). *An open platform for developing, deploying, and accessing planetary-scale services*. Retrieved 31 March, 2016, from https://www.planet-lab.org/

Price, D., & Tucker, A. (2004). Solaris Zones: Operating System Support for Consolidating Commercial Workloads.*Proceedings of the 18th USENIX Conference on System Administration (LISA 2004)* (pp. 241-254).

Reiss, C., Wilkes, J., & Hellerstein, J. L. (2011). *Google cluster-usage traces: format+ schema*. Academic Press.

Roadef. (2016). *Challenge ROADEF/EURO 2012: Machine Reassignment*. Retrieved 10 May, 2016, from http://challenge.roadef.org/2012/en/index.php

Rocket. (2016). *Welcome to Rocket Diagram's documentation! — Rocket Diagram 0.1.0 documentation*. Retrieved 10 May, 2016, from https://rocketdiagram.readthedocs.io/en/latest/

Rolia, J., Andrzejak, A., & Arlitt, M. (2003). *Automating Enterprise Application Placement in Resource Utilities*. Self-Managing Distributed Systems. doi:10.1007/978-3-540-39671-0_11

Rosen, R. (2013, May). Resource management: Linux kernel Namespaces and cgroups. *Haifux*.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*, 53–65. doi:10.1016/0377-0427(87)90125-7

Solis Moreno, I., Garraghan, P., Townend, P., & Xu, J. (2014). Analysis, Modeling and Simulation of Workload Patterns in a Large-Scale Utility Cloud. *IEEE Transactions on Cloud Computing*, *2*(2), 208–221. doi:10.1109/TCC.2014.2314661

Spicuglia, S., Chen, L. Y., Birke, R., & Binder, W. (2015). Optimizing capacity allocation for big data applications in cloud datacenters.*Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM 2015)* (pp. 511-517). doi:10.1109/INM.2015.7140330

Svärd, P., Hudzia, B., Tordsson, J., & Elmroth, E. (2011). Evaluation of delta compression techniques for efficient live migration of large virtual machines. *ACM Sigplan Notices*, *46*(7), 111–120. doi:10.1145/2007477.1952698

Tchana, A., Palma, N., Safieddine, I., Hagimont, D., Diot, B., & Vuillerme, N. (2015). Software Consolidation as an Efficient Energy and Cost Saving Solution for a SaaS/PaaS Cloud Model.*Proceedings of the 2015 European Conference on Parallel Processing (Euro-Par 2015)* (pp. 305-316). doi:10.1007/978-3-662-48096-0_24

Tomas, L., Klein, C., Tordsson, J., & Hernandez-Rodriguez, F. (2014). The Straw that Broke the Camel's Back: Safe Cloud Overbooking with Application Brownout.*Proceedings of the 2014 International Conference on Cloud and Autonomic Computing (ICCAC 2014)* (pp. 151-160). doi:10.1109/ICCAC.2014.10

Tomás, L., & Tordsson, J. (2014). An autonomic approach to risk-aware data center overbooking. *IEEE Transactions on Cloud Computing*, *2*(3), 292–305. doi:10.1109/TCC.2014.2326166

Topcuoglu, H., Hariri, S., & Wu, M.-Y. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Transactions on Parallel and Distributed Systems*, *13*(3), 260–274. doi:10.1109/71.993206

Urgaonkar, R., Kozat, U. C., Igarashi, K., & Neely, M. J. (2010). Dynamic resource allocation and power management in virtualized data centers.*Proceedings of the 2010 IEEE Network Operations and Management Symposium (NOMS 2010)* (pp. 479-486). doi:10.1109/NOMS.2010.5488484

Verma, A., Dasgupta, G., Nayak, T. K., De, P., & Kothari, R. (2009). Server Workload Analysis for Power Minimization Using Consolidation.*Proceedings of the 2009 Conference on USENIX Annual Technical Conference* (pp. 28-28).

von Laszewski, G., Wang, L., Younge, A. J., & He, X. (2009). Power-aware scheduling of virtual machines in DVFS-enabled clusters.*Proceedings of the 2009 IEEE International Conference on Cluster Computing and Workshops (CLUSTER 2009).* (pp. 1-10). doi:10.1109/CLUSTR.2009.5289182

VServer. (2016). *Overview - Linux-VServer.* Retrieved from http://linux-vserver.org/Overview

Wang, D., Ren, C., Govindan, S., Sivasubramaniam, A., Urgaonkar, B., Kansal, A., & Vaid, K. (2013). ACE: abstracting, characterizing and exploiting peaks and valleys in datacenter power consumption. *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems* (pp. 333-334). doi:10.1145/2465529.2465536

Wang, L., Tao, J., von Laszewski, G., & Chen, D. (2010). *Power Aware Scheduling for Parallel Tasks via Task Clustering*. Paper presented at the IEEE 16th International Conference on Parallel and Distributed Systems (ICPADS 2010). doi:10.1109/ICPADS.2010.128

Yahoo. (2010, November). *Yahoo! Expands Its M45 Cloud Computing Initiative*. Retrieved November, 2010, from https://yodel.yahoo.com/blogs/product-news/yahoo-expands-m45-cloud-computing-initiative-5065.html

Yaqub, E., Yahyapour, R., Wieder, P., Jehangiri, A. I., Lu, K., & Kotsokalis, C. (2014). Metaheuristics-Based Planning and Optimization for SLA-Aware Resource Management in PaaS Clouds.*Proceedings of the 7th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2014)* (pp. 288-297). doi:10.1109/UCC.2014.38

Yogamangalam, R., & Sriram, V. S. (2013). A Review on Security Issues in Cloud Computing. *Journal of Artificial Intelligence*, *6*(1), 1–7. doi:10.3923/jai.2013.1.7

Zhang, X., Tune, E., Hagmann, R., Jnagal, R., Gokhale, V., & Wilkes, J. (2013). CPI2: CPU Performance Isolation for Shared Compute Clusters.*Proceedings of the 8th ACM European Conference on Computer Systems (EuroSys '13)* (pp. 379-391). doi:10.1145/2465351.2465388

ENDNOTES

[1] The scheduling class shows how sensitive a task is to latency. In Google traces the scheduling class is an integer number between 0 and 3 and the higher the scheduling class is, the most latency sensitive the task

[2] The input or drive workload is the workload under which the performance of the system is tested.